

A L^AT_EX Package of utility macros ^{*†}

Arthur Ogawa [‡]

January 18, 2019

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

The distribution point for this work is journals.aps.org/revtex, which contains prebuilt runtime files, documentation, and full source, ready to add to a TDS-compliant T_EX installation.

The `ltxutil` package was commissioned by the American Physical Society and is distributed under the terms of the L^AT_EX Project Public License 1.3c, the same license under which all the portions of L^AT_EX itself are 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 pdftex and Adobe Acrobat Reader or equivalent.

To install, retrieve the distribution, unpack it into a directory on the target computer, and move the file `ltxutil.sty` into a location in your filesystem where it will be found by L^AT_EX.

To use, read the user documentation `ltxutil.pdf`.

Contents

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

^{*}This file has version number 4.2c, last revised 2019/01/18/14:29:48.

[†]Version 4.2c © 2019 American Physical Society

[‡]mailto:arthur_ogawa@sbcglobal.net

3	The driver module <code>doc</code>	5
3.1	The Preamble	5
3.1.1	Docstrip and info directives	5
3.2	The “Read Me” File	6
3.3	The Document Body	9
4	Using this package	9
4.1	Invoking the package	9
5	Compatibility with L^AT_EX’s Required Packages	10
5.1	<code>array</code>	10
5.2	<code>longtable</code>	10
6	Implementation of package	10
6.1	Beginning of the <code>package</code> <code>DOCSTRIP</code> module	10
6.2	Banner and beginning of the <code>kernel</code> <code>DOCSTRIP</code> module	11
6.3	Errors and warnings	11
6.4	New Tools	11
6.5	Boolean Control	13
6.6	Begin Document Structure	15
6.7	Class Extensions	18
6.8	Type Tools	19
6.9	Display Math	20
6.10	Footnotes	22
6.11	Floats	27
6.11.1	Usage notes	27
6.11.2	Robustifying fragile commands	29
6.11.3	Preparing for the <code>hyperref</code> package	29
6.11.4	Footnotes within floats, unfloating floats, float font	30
6.11.5	Writing floats out to a file	32
6.12	Counters	36
6.13	Customization of Sections	36
6.14	Patch the <code>tabular</code> and <code>array</code> Environments	40
6.15	Repair other broken parts of L ^A T _E X	62
6.16	Syntax	63
6.17	Auto-indented Contents	63
6.18	Lists	67
6.19	Hypertext capabilities	67
6.20	End of the <code>kernel</code> <code>DOCSTRIP</code> module	70

1 Processing Instructions

The package file `ltxutil.sty` is generated from this file, `ltxutil.dtx`, using the `DOCSTRIP` facility of L^AT_EX via `tex ltxutil.dtx` (Note: do *not* use L^AT_EX for this task). The typeset documentation that you are now reading is generated from

the same file by typesetting it with \LaTeX or `pdftex` via `latex ltxutil.dtx` or `pdflatex ltxutil.dtx`.

1.1 Build Instructions

You may bootstrap this suite of files solely from `ltxutil.dtx`. Prepare by installing $\LaTeX 2_\epsilon$ (and either `tex` or `pdftex`) on your computer, then carry out the following steps:

1. Within an otherwise empty directory, typeset `ltxutil.dtx` with \LaTeX or `pdflatex`; you will obtain the typeset documentation you are now reading, along with the file `README-LTXUTIL`.

Note: you will have to run \LaTeX , then `makeindex -s gind.ist ltxutil.idx`, then `makeindex -s gglo.ist -o ltxutil.gls ltxutil.glo`, then \LaTeX again in order to obtain a valid index and table of contents.

2. Now typeset `ltxutil.dtx` with \TeX (not \LaTeX), thereby generating the package file `ltxutil.sty`.
3. Install the following files into indicated locations within your TDS-compliant `texmf` tree (you may need root access):

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

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

4. Run `mktexlsr` on `$TEXMF/` (you may need root access).
5. Build and installation are now complete; now put a `\usepackage{ltxutil}` in your document preamble!

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.

```
%ltxutil.dtx
%
```

1.3.2 Generated by latex ltxutil.dtx

Typesetting the source file under pdf_latex generates the readme and the documentation.

```
%README-LTXUTIL ltxutil.pdf
%
```

1.3.3 Generated by tex ltxutil.dtx

Typesetting this file with T_EX generates the package file.

```
%ltxutil.sty
%
```

1.3.4 Auxiliary

The following are auxiliary files generated in the course of running L^AT_EX:

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

2 Code common to all modules

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

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

```
4 %<kernel&!package&!doc>\typeout{%
5 %<*package|doc>
6 \ProvidesFile{%
7 %</package|doc>
8 %<*kernel|package|doc>
9 ltxutil%
10 %</kernel|package|doc>
11 %<*doc>
12 .dtx%
13 %</doc>
14 %<package>.sty%
15 %<*package|doc>
16 }%
17 %</package|doc>
```

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. Give credit where due.

```
18 %<*doc|package|kernel>
19 %<version>
20 [2019/01/18/14:29:48 4.2c utilities package (portions licensed from W. E. Baxter web at supers
21 %</doc|package|kernel>
22 %<kernel&!package&!doc>}%
```

3 The driver module doc

This module, consisting of the present section, typesets the programmer's documentation, generating the `README-LTXUTIL` 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.

```
23 %<*doc>
```

3.1 The Preamble

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

```
24 \documentclass{ltxdoc}
25 \RequirePackage{ltxdocext}%
26 \let\url\undefined
27 \RequirePackage[colorlinks=true,linkcolor=blue]{hyperref}%
28 %\expandafter\ifx\cename package@font\endcsname\@undefined\else
29 % \expandafter\RequirePackage\expandafter{\cename package@font\endcsname}%
30 %\fi
31 \CodelineIndex\EnableCrossrefs % makeindex -s gind.ist ltxutil
32 \RecordChanges % makeindex -s gglo.ist -o ltxutil.gls ltxutil.glo
```

3.1.1 Docstrip and info directives

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

```
33 \setcounter{StandardModuleDepth}{1}
```

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

```
34 \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 out these procedures begin the programmer’s manual.

```
35 \begin{filecontents*}{README-LTXUTIL}
36 \title{%
37 A \LaTeX\ Package of utility macros%
38 \thanks{%
39 This file has version number \fileversion,
40 last revised \filedate.%
41 }%
42 \thanks{%
43 Version \fileversion\ \copyright\ 2019 American Physical Society
44 }%
45 }%
46 \author{%
47 Arthur Ogawa%
48 \thanks{\texttt{mailto:arthur\_ogawa at sbcglobal.net}}}%
49 }%
50 %\iffalse
51 % For version number and date,
52 % search on "\fileversion" in the .dtx file,
53 % or see the end of the README-LTXUTIL file.
54 %\fi
55 \maketitle
56
57 This file embodies the \classname{ltxutil} package,
58 the implementation and its user documentation.
59
60 The distribution point for this work is
61 \url{journals.aps.org/revtex},
62 which contains prebuilt runtime files, documentation, and full source,
63 ready to add to a TDS-compliant \TeX\ installation.
64
65 The \classname{ltxutil} package was commissioned by the American Physical Society
66 and is distributed under the terms of the \LaTeX\ Project Public License 1.3c,
67 the same license under which all the portions of \LaTeX\ itself are distributed.
68 Please see \url{http://ctan.tug.org/macros/latex/base/lppl.txt} for details.
69
70 To use this document class, you must have a working
71 \TeX\ installation equipped with \LaTeXe\
```

72 and possibly pdftex and Adobe Acrobat Reader or equivalent.
73
74 To install, retrieve the distribution,
75 unpack it into a directory on the target computer,
76 and move the file `\file{ltxutil.sty}`
77 into a location in your filesystem where it will be found by `\LaTeX`.
78
79 To use, read the user documentation `\file{ltxutil.pdf}`.
80
81 `\tableofcontents`
82
83 `\section{Processing Instructions}`
84
85 The package file `\file{ltxutil.sty}`
86 is generated from this file, `\file{ltxutil.dtx}`,
87 using the `{\sc docstrip}` facility of `\LaTeX`
88 via `|tex ltxutil.dtx|` (Note: do `\emph{not}` use `\LaTeX` for this task).
89 The typeset documentation that you are now reading is generated from
90 the same file by typesetting it with `\LaTeX` or `pdftex`
91 via `|latex ltxutil.dtx|` or `|pdflatex ltxutil.dtx|`.
92
93 `\subsection{Build Instructions}`
94
95 You may bootstrap this suite of files solely from `\file{ltxutil.dtx}`.
96 Prepare by installing `\LaTeXe` (and either `tex` or `pdftex`) on your computer,
97 then carry out the following steps:
98 `\begin{enumerate}`
99 `\item`
100 Within an otherwise empty directory,
101 typeset `\file{ltxutil.dtx}` with `\LaTeX` or `pdflatex`;
102 you will obtain the typeset documentation you are now reading,
103 along with the file `\file{README-LTXUTIL}`.
104
105 Note: you will have to run `\LaTeX`, then
106 `\file{makeindex} \texttt{-s gind.ist ltxutil.idx}`, then
107 `\file{makeindex} \texttt{-s gglo.ist -o ltxutil.gls ltxutil.glo}`, then
108 `\LaTeX` again in order to obtain a valid index and table of contents.
109 `\item`
110 Now typeset `\file{ltxutil.dtx}` with `\TeX` (not `\LaTeX`),
111 thereby generating the package file `\file{ltxutil.sty}`.
112 `\item`
113 Install the following files into indicated locations within your
114 TDS-compliant `\texttt{texmf}` tree (you may need root access):
115 `\begin{itemize}`
116 `\item`
117 `\file{$TEXMF/}\file{tex/}\file{latex/}\file{revtex/}\classname{ltxutil.sty}`
118 `\item`
119 `\file{$TEXMF/}\file{source/}\file{latex/}\file{revtex/}\classname{ltxutil.dtx}`
120 `\item`
121 `\file{$TEXMF/}\file{doc/}\file{latex/}\file{revtex/}\classname{ltxutil.pdf}`

```

122 \end{itemize}
123 where \file{$TEXMF/} stands for \file{texmf-local/}, or some other \texttt{texmf} tree
124 in your installation.
125 \item
126 Run \texttt{mktexlsr} on \file{$TEXMF/} (you may need root access).
127 \item
128 Build and installation are now complete;
129 now put a \cmd\usepackage\texttt{\{ltxutil\}} in your document preamble!
130 \end{enumerate}
131
132 \subsection{Change Log}
133 \changes{4.0b}{1999/06/20}{AO: Fixed spurious \texttt{CR} and (return) characters in output fil
134 \changes{4.0b}{1999/06/20}{AO: Removed superfluous \cs{def}s, changed to using \cs{floats@sw} a
135 \changes{4.0b}{1999/06/20}{only execute if there really were floats of the given type}
136 \changes{4.0b}{1999/06/20}{Support the hack with \cs{prepdef}, and delay until \cs{AtBeginDocum
137 \changes{4.0c}{1999/11/13}{(AO, 110) Install hooks for endfloats processing}
138 \changes{4.0c}{1999/11/13}{(AO, 116) Hyperref compatibility}
139 \changes{4.0c}{1999/11/13}{(AO, 130) Interference from array package}
140 \changes{4.0c}{1999/11/13}{*-form mandates pagebreak at each float; only print section head if
141 \changes{4.0d}{2000/04/10}{(AO, 127) Floats placed [h] to allow page breaks}
142 \changes{4.0d}{2000/04/10}{(AO, 174) kernel fix}
143 \changes{4.0d}{2000/05/19}{(AO, 224) Hyperref compatibility.}
144 \changes{4.0d}{2000/05/23}{Allow things to break over pages by setting array@default.}
145 \changes{4.0e}{2000/11/16}{(AO, 221) Remove samepage command from @xfloat@prep: If the float ca
146 \changes{4.0f}{2001/07/13}{(AO, 404) Hyperref compatibility}
147 \changes{4.1a}{2008/01/19}{(AO, 459) do not assume \cs{classname} is defined}%
148 \changes{4.1a}{2008/01/19}{(AO, 461) Change the csname from \cs{@dotsep} to \cs{ltxu@dotsep}. T
149 \changes{4.1a}{2008/01/19}{(AO, 475) I had not properly reproduced the LaTeX macro \cs{eqnarray
150 \changes{4.1a}{2008/01/19}{(AO, 479) Per: Dylan Thurston<dpt at math.harvard.edu>}%
151 \changes{4.1a}{2008/06/30}{(AO) Make \cs{addtocontents} a \cs{long} \cs{def}; gobble up \cs{foo
152 \changes{4.1a}{2008/06/30}{(AO) Remove code that avoided changes to \cs{@xfootnotemark}}%
153 \changes{4.1a}{2008/06/30}{(AO, 438) Complete rewrite of footnote macros.}
154 \changes{4.1a}{2008/07/07}{\cs{@xfloat@prep} calls \cs{ltxu@footnote@pop} to restore the origina
155 \changes{4.1a}{2008/08/12}{\cs{class@documenthook} is the last \cs{AtBeginDocument} token now}
156 \changes{4.1a}{2008/08/12}{Class extension mechanism \cs{@pushfilename@ltx} and \cs{@p@pfilenam
157 \changes{4.1a}{2008/08/12}{Class extension mechanism \cs{class@extension}, \cs{class@extensionf
158 \changes{4.1a}{2008/08/12}{Get rid of \cs{set@typesize@hook} \cs{set@pica@hook} and the \cs{nor
159 \changes{4.1b}{2008/08/12}{(AO, 487) Support for video figures and the \cs{setfloatlink} comman
160 \changes{4.1b}{2008/08/12}{(AO, 505) try to accommodate \classname{colortbl}.}
161 \changes{4.1b}{2008/08/12}{Acquire \classname{hyperref} saveoire}
162 \changes{4.1b}{2008/08/12}{Default assignment of \cs{float@sw} now, not at \cs{AtBeginDocument}
163 \changes{4.1b}{2008/08/12}{If class option \classoption{lengthcheck} is in effect, log the heig
164 \changes{4.1b}{2008/08/12}{No need to protect against undefined \cs{float@sw}}
165 \changes{4.1b}{2008/08/12}{Patch the array package even later: after all package patches go in.
166 \changes{4.1b}{2008/08/12}{Refine toc processing: provide default.}%
167 \changes{4.1b}{2008/08/12}{Tally and log the height of a float class}
168 \changes{4.1d}{2009/03/27}{(AO, 511) Compatability with lineno.sty's erroneous way of detecting
169 \changes{4.1f}{2009/07/07}{(AO, 515) Hook for setting the font of a footnote}
170 \changes{4.1f}{2009/07/10}{(AO, 518) Tally register overflow when locument is long}
171 \changes{4.1g}{2009/10/06}{(AO, 532) Both arguments of \cs{href} get sanitized}%

```



```

172 \changes{4.1g}{2009/10/07}{(AO, 525) Remove phantom paragraph above display math that is given
173 \changes{4.1g}{2009/10/07}{(AO, 539) Use of double-backslash in argument of \cs{section} gives
174 \changes{4.1n}{2009/12/05}{(AO, 569) Use of \classname{hyperref} interferes with column balanci
175 \changes{4.1n}{2009/12/06}{(AO) Incorporate change to ltmiscen.dtx v1.1i 2000/05/19}%
176 \changes{4.1n}{2009/12/09}{(AO, 569) execute \classname{atveryend}'s \cs{Call@AfterLastShipout}
177 \changes{4.1n}{2009/12/13}{(AO, 574) protect against \classname{lineno.sty}, which forces a vis
178 \changes{4.1n}{2010/01/02}{(AO, 571) Interface \cs{set@footnotewidth} for determining the set w
179 \changes{4.1n}{2010/01/02}{(AO, 571) allow split after last line of footnote}%
180 \changes{4.1n}{2010/01/06}{(AO, 572) title block footnotes numbered independently from body foo
181 \changes{4.1p}{2010/02/24}{(AO, 582) A patch of \classname{hyperref.sty} to provide backward co
182 \changes{4.2a}{2017/11/21}{(MD) Use updated best practice to use https and doi.org}%
183 \changes{4.2a}{2018/12/12}{(MD) Updated name of README file and use standard fonts when typeset
184
185 \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.

```

186 \begin{document}%
187 \expandafter\DocInput\expandafter{\jobname.dtx}%
188 \end{document}
189 %</doc>

```

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{ltxutil}%
%\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}%
%\RequirePackage{ltxutil}%
%\LoadClass{book}%
%(class customization commands)
%\endinput

```

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.

5 Compatibility with \LaTeX 's Required Packages

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

The `ltxutil` package will be incompatible with any package that redefines any of the kernel macros that `ltxutil` patches—if that package is loaded *after* `ltxutil`. This means that for greatest compatibility, `ltxutil` should be loaded *after*, say, `ftnright`, which overwrites \LaTeX 's kernel procedures `\@outputdblcol`, `\@startcolumn`, and `\@makecol`.

Hereinafter follows some notes on specific \LaTeX packages.

5.1 `array`

This package alters the way tabular environments are done, therefore it could run afoul of the \LaTeX “required” package `array` or any package that calls for it to be loaded. However, this package has provisions for remaining compatible with `array`. So long as the version of `array` that is used with this package has the appropriate meanings for the procedures it overwrites, all should be well.

5.2 `longtable`

David Carlisle's `longtable` package modifies both the \LaTeX kernel and the `array` package. This package must therefore alter `\LT@array`. For now, that job is handled by `ltxgrid`.

6 Implementation of package

Special acknowledgment: this package uses concepts pioneered and first realized by William Baxter (<mailto:web@superscript.com>) in his SuperScript line of commercial typesetting tools, and which are used here with his permission.

6.1 Beginning of the package `DOCSTRIP` module

```
190 %<*package>
191 \def\package@name{ltxutil}%
192 \expandafter\PackageInfo\expandafter{\package@name}{%
193 Utility macros for \protect\LaTeXe,
194 by A. Ogawa (arthur_ogawa at sbcglobal.net)%
195 }%
196 %</package>
```

6.2 Banner and beginning of the kernel DOCSTRIP module

```
197 %<*kernel>
```

6.3 Errors and warnings

`\class@err` A few shorthands for Class messages. Your document class should define
`\class@warn` `\class@name`.

```
\class@info 198 \def\class@err#1{\ClassError{\class@name}{#1}\@eha}%
199 \def\class@warn#1{\ClassWarningNoLine{\class@name}{#1}}%
200 \def\class@info#1{\ClassInfo{\class@name}{#1}}%
201 \def\obsolete@command#1{%
202 \class@warn@end{Command \string#1\space is obsolete.^^JPlease remove from your document}%
203 \global\let#1\@empty
204 #1%
205 }%
206 \def\replace@command#1#2{%
207 \class@warn@end{Command \string#1\space is obsolete;^^JUse \string#2\space instead}%
208 \global\let#1#2%
209 #1%
210 }%
211 \def\replace@environment#1#2{%
212 \class@warn@end{Environment #1 is obsolete;^^JUse #2 instead}%
213 \glet@environment{#1}{#2}%
214 \@nameuse{#1}%
215 }%
216 \def\incompatible@package#1{%
217 \@ifpackageloaded{#1}{%
218 \def\@tempa[I cannot continue. You must remove the \string\usepackage\ statement that caused
219 \ClassError{\class@name}{The #1 package cannot be used with \class@name}%
220 \@tempa\stop
221 }{%
222 \class@info{#1 was not loaded (OK!)}%
223 }%
224 }%
225 \def\class@warn@end#1{%
226 \gapdef\class@enddocumenthook{\class@warn{#1}}%
227 }%
```

Give `\class@name` a meaning if it does not already have one.

```
228 \ifx\undefined\class@name
229 \def\class@name{ltxutil}%
230 \class@warn{You should define the class name before reading in this package. Using default}%
231 \fi
```

6.4 New Tools

```
\t@
232 \def\t@{to}%
```

```

\dimen@iii
233 \dimendef\dimen@iii\thr@@

\halign@
234 \def\halign@{\halign\t@}%

\fur Analogous to \@ne, \tw@, and \thr@@.
235 \chardef\fur=4\relax
236 \chardef\cat@letter=11\relax
237 \chardef\other=12\relax

\let@environment The directive \let@environment takes care of a common programming idiom
\glet@environment whereby one environment is made a synonym for another.
238 \def\let@environment#1#2{%
239 \expandafter\let
240 \csname#1\expandafter\endcsname\csname#2\endcsname
241 \expandafter\let
242 \csname end#1\expandafter\endcsname\csname end#2\endcsname
243 }%
244 \def\glet@environment#1#2{%
245 \global\expandafter\let
246 \csname#1\expandafter\endcsname\csname#2\endcsname
247 \global\expandafter\let
248 \csname end#1\expandafter\endcsname\csname end#2\endcsname
249 }%

\tracingplain The command \tracingplain causes TEX's tracing parameters to return to the
values set by default. This command is sometimes useful when you have said
\tracingall somewhere and want to restore. The \traceoutput command
causes \tracingoutput diagnostics upon \shipout.
250 \newcommand\tracingplain{%
251 \tracingonline\z@\tracingcommands\z@\tracingstats\z@
252 \tracingpages\z@\tracingoutput\z@\tracinglostchars\@ne
253 \tracingmacros\z@\tracingparagraphs\z@\tracingrestores\z@
254 \showboxbreadth5\showboxdepth3\relax %\errorstopmode
255 }%
256 \newcommand\traceoutput{%
257 \appdef\@resetactivechars{\showoutput}%
258 }%

\say The commands \say and \saythe cause diagnostic messages in the TEX log that
\saythe give the value of a control sequence name or a register respectively.
259 \newcommand\say[1]{\typeout{<\noexpand#1=\meaning#1>}}%
260 \newcommand\saythe[1]{\typeout{<\noexpand#1=\the#1>}}%

\fullinterlineskip Resets the \prevdepth so that the full amount of \baselineskip glue will be
inserted by the \baselineskip mechanism. Can be invoked just after a \hrule
to undo its default suppression of base line skip.
261 \def\fullinterlineskip{\prevdepth\z@}%

```

```

\count@i
\count@ii 262 \countdef\count@i\@ne
           263 \countdef\count@ii\tw@

```

6.5 Boolean Control

We introduce just enough of the Boolean calculus for \TeX . Alan Jeffrey was the pioneer here, with an article in TUGboat (Vol. 11, No. 2, page 237). This implementation owes a debt to William Baxter (web at superscript.com). See articles by Baxter and Ogawa in the proceedings of the 1994 TUG meeting, TUGboat Vol. 15, No. 3.

```

\prepdef Provide the capability of performing head- and tail patches. The procedure
\appdef  \prepdef prepends to the given macro the tokens specified in its second argument.
\gappdef Likewise for \appdef, except that it appends. Note that the first 10 toks registers
           are utility registers, and we simply make a control sequence name, \toks@ii, for
           one of them.

```

```

264 \long\def\prepdef#1#2{%
265   \@ifxundefined#1{\toks@{}}{\toks@\expandafter{#1}}%
266   \toks@ii{#2}%
267   \edef#1{\the\toks@ii\the\toks@}%
268 }%
269 \long\def\appdef#1#2{%
270   \@ifxundefined#1{\toks@{}}{\toks@\expandafter{#1}}%
271   \toks@ii{#2}%
272   \edef#1{\the\toks@\the\toks@ii}%
273 }%
274 \long\def\gappdef#1#2{%
275   \@ifxundefined#1{\toks@{}}{\toks@\expandafter{#1}}%
276   \toks@ii{#2}%
277   \global\edef#1{\the\toks@\the\toks@ii}%
278 }%
279 \long\def\appdef@val#1#2{%
280   \appdef#1{#2}%
281 }%
282 \long\def\appdef@e#1#2{%
283   \expandafter\appdef
284   \expandafter#1%
285   \expandafter{#2}%
286 }%
287 \long\def\appdef@eval#1#2{%
288   \expandafter\appdef@val
289   \expandafter#1%
290   \expandafter{#2}%
291 }%
292 \toksdef\toks@ii=\tw@

```

```

\@ifxundefined Certain utility procedures use \@ifxundefined, which is defined here in terms
\@ifnotrelax   of \@ifx. Others use \@ifnotrelax, namely when the control sequence name is
\@argswap
\@argswap@val

```

manufactured by the use of `\csname`.

The procedures `\@argswapand` `\@argswap@val` are used to facilitate control of expansion.

```
293 \long\def\@ifxundefined#1{\@ifx{\undefined#1}}%
294 \long\def\@ifnotrelax#1#2#3{\@ifx{\relax#1}{#3}{#2}}%
295 \long\def\@argswap#1#2{#2#1}%
296 \long\def\@argswap@val#1#2{#2{#1}}%
297 \def\@ifxundefined@cs#1{\expandafter\@ifx\expandafter{\csname#1\endcsname\relax}}%
```

`\@boolean` In order to define `\@ifx`, we first must create the “defining word” (term taken from our Forth vocabulary) `\@boole@def`, which employs `\@boolean` to do its job.

```
298 \def\@boolean#1#2{%
299   \long\def#1{%
300     #2% \if<something>
301     \expandafter\true@sw
302   \else
303     \expandafter\false@sw
304   \fi
305 }%
306 }%
307 \def\@boole@def#1#\@boolean{#1}}% Implicit #2
```

`\@booleantrue` `\@booleanfalse` The procedures `\@booleantrue` and `\@booleanfalse` are assignment operators for Boolean flags.

```
308 \def\@booleantrue#1{\let#1\true@sw}%
309 \def\@booleanfalse#1{\let#1\false@sw}%
```

`\@ifx` We can now invoke the defining word to create the procedures `\@ifx` and friends. Compatibility Note: earlier versions of this package defined a procedure `\@ifempty`. However, for compatibility with AMS \LaTeX , we must avoid the following three names: `\@ifempty`, `\@xifempty`, and `\@ifnotempty`.

```
\@ifcat
\@ifdim
\@ifeof
\@ifhbox
\@ifhmode
\@ifinner
\@ifmmode
\@ifnum
\@ifodd
\@ifvbox
\@ifvmode
\@ifvoid
310 \@boole@def\@ifx#1{\ifx#1}%
311 \@boole@def\@ifx@empty#1{\ifx\@empty#1}%
312 \@boole@def\@ifempty#1{\if!#1!}%
313 %\@boole@def\@ifsw#1{\csname if#1\endcsname}%
314 \def\@ifsw#1#2{#1\expandafter\true@sw\else\expandafter\false@sw#2}%
315 \@boole@def\@ifdim#1{\ifdim#1}%
316 \@boole@def\@ifeof#1{\ifeof#1}%
317 \@boole@def\@ifhbox#1{\ifhbox#1}%
318 \@boole@def\@ifhmode{\ifhmode}%
319 \@boole@def\@ifinner{\ifinner}%
320 \@boole@def\@ifmmode{\ifmmode}%
321 \@boole@def\@ifnum#1{\ifnum#1}%
322 \@boole@def\@ifodd#1{\ifodd#1}%
323 \@boole@def\@ifvbox#1{\ifvbox#1}%
324 \@boole@def\@ifvmode{\ifvmode}%
325 \@boole@def\@ifvoid#1{\ifvoid#1}%
```

`\true@sw` Note that when a Boolean operator expands, it employs two macros that act as selectors, defined here.

```
326 \long\def\true@sw#1#2{#1}%
327 \long\def>false@sw#1#2{#2}%
```

`\loopuntil` Loop control using the Boolean idiom. Superior to `\loop... \repeat` because these can be nested. The tail of the argument must have a Boolean predicate.

```
328 \long\def\loopuntil#1{#1}{\loopuntil{#1}}}%
329 \long\def\loopwhile#1{#1}{\loopwhile{#1}}{}}%
```

`\@provide` A defining word that refuses to clobber a prior meaning.

```
330 \def\@provide#1{%
331 \ifx{\undefined#1}{\true@sw}{\ifx{\relax#1}{\true@sw}{\false@sw}}}%
332 {\def#1}{\def\j@nk}%
333 }%
```

6.6 Begin Document Structure

The standard L^AT_EX mechanism `\AtBeginDocument` is inadequate because the `\vsize` is bound much too early. We supply here a mechanism whereby decisions about the page layout can be deferred until `\AtBeginDocument` time.

The problem we are working around is that the `\AtBeginDocument` hook in `\document` appears long after the calculation of `\vsize` and `\hsize`, that is, L^AT_EX provides no mechanism for deferring the decision about the page grid until `\AtBeginDocument` time. We fix things by prepending a hook at the very beginning of `\document`.

As it turns out, though, it appears feasible to simply invoke the desired column grid command at `\AtBeginDocument` time, since the MVL has nothing in it at that time that would be problematical.

The facility depends on the stability of this part of L^AT_EX's kernel code (the first token of `\document`), which could change, you see. But considering that L^AT_EX is at this point essentially stagnant once more, we risk it.

`\document` We begin by installing hooks into `\document` that we will manage ourselves. First, we do as `\document` does: end the group begun by `\begin`. Last, we conclude our shenanigans by absorbing the first token of the expansion of `\document`, which we assume to be `\endgroup`.

```
334 \prepdef\document{%
335 \endgroup
336 \document@inithook
337 \true@sw{}}%
338 }%
```

`\document@inithook` To use, simply `\appdef\document@inithook{<your tokens here>}`.

```
339 \let\document@inithook\@empty
```

`\class@documenthook` We install the last `\AtBeginDocument` hook, namely the procedure `\class@documenthook`.
`\class@enddocumenthook` Within the document class, we will use this hook exclusively, so as to avoid interference from other packages. Similarly with `\class@enddocumenthook`, installed via `\AtEndDocument`.

A document class using this package should do as this package does and just say, `\appdef \class@documenthook` instead of `\AtBeginDocument`, and `\appdef \class@enddocumenthook` instead of `\AtEndDocument`.

```
340 \appdef\document@inithook{%
341 \AtBeginDocument{\class@documenthook}%
342 }%
343 \AtEndDocument{%
344 \class@enddocumenthook
345 }%
346 \let\class@documenthook\@empty
347 \let\class@enddocumenthook\@empty
```

`\enddocument` The standard L^AT_EX `\end{document}` processing is a potential problem, particularly when the output routine has been changed by `ltxgrid`. We separate out the procedure that checks the auxiliary file at the end of the job so that later it can be called from the safety of the output routine. We will do this to ensure that the `\@mainaux` stream is not closed until the last page of the job is shipped out, and that can only be done by coordinating with the output routine.

```
348 \def\enddocument{%
The following line from ltxutil.dtx ‘resets \AtEndDocument for latex/3060’.
349 \let\AtEndDocument\@firstofone
350 \@enddocumenthook
351 \@checkend{document}}%
```

The `\clear@document` statement ends the current page (we must guarantee no further shipouts), then executes all cleanup procedures that must occur only after the last shipout. Clients will queue up their procedures via `\AfterLastShipout`, if it exists, otherwise by doing `\appdef\clear@document`.

```
352 \clear@document
We are very close to ending the TEX run, now.
353 \check@aux
354 \deadcycles\z@
355 \@@end
356 }%
357 \def\check@aux{\do@check@aux}%
358 \def\do@check@aux{%
359 \@ifsw@if@filesw@fi{%
360 \immediate\closeout\@mainaux
361 \let\@setckpt\@gobbletwo
362 \let\@newl@bel\@testdef
363 \@tempwafalse
364 \makeatletter
```



```

365 \input\jobname.aux\relax
366 }{}%
367 \@dofilelist
368 \@ifdim\font@submax >\fontsubfuzz\relax}{%
369 \font@warning{%
370   Size substitutions with differences\MessageBreak
371   up to \font@submax\space have occurred.\@gobbletwo
372 }%
373 }{}%
374 \@defaultsubs
375 \@refundefined
376 \@if@sw@if@filesw\fi{%
377 \@ifx{\@multiplelabels\relax}{%
378 \@if@sw@if@tempswa\fi{%
379 \latex@warning@no@line{%
380   Label(s) may have changed.
381   Rerun to get cross-references right
382 }%
383 }{}%
384 }{%
385 \multiplelabels
386 }%
387 }{}%
388 }%

```

`\clear@document` The procedure `\clear@document` is responsible for flushing out the last page of the document, if not already done. The procedure then executes those procedures that must wait for execution until after the last page is shipped out. Clients of `ltxutil`, such as `ltxgrid` and `revtex4` will queue these procedures up via `\AfterLastShipout`, if it exists, otherwise by doing `\appdef\clear@document`.

The command `\Call@AfterLastShipout` is provided by Heiko Oberdiek's `atveryend` package. This package is compatible with `ltxutil`.

Note on compatibility with `atveryend`: we arrange for `\Call@AfterLastShipout` to be called from the safety of the output routine, thereby ensuring that all of the procedures queued up by that package's `\AfterLastShipout` are executed at the right time. We also ensure that `\Call@AfterLastShipout` has a default definition, in case the package was never loaded.

```

389 \def\clear@document{%
390 \clearpage
391 \do@output@cclv{%
392 \Call@AfterLastShipout
393 }%
394 }%
395 \appdef\class@documenthook{%
396 \providecommand\Call@AfterLastShipout{}}%
397 }%

```

6.7 Class Extensions

The L^AT_EX procedure `\@onefilewithoptions` is the vehicle for reading in a L^AT_EX class or package. The APS RevTeX class implements the use of what are called “substyles”, actually extensions to the class itself. Any document class can do likewise.

`\class@extension` A procedure similar to L^AT_EX’s `\@onefilewithoptions`, but as an extension to
`\class@extensionfile` the current document class.
`\class@ext@hook` Read in the given file as if it were a document class file. Usage: `\class@extensionfile`
`{\class}` `\@extension`, where `\class` is a file (similar to `aps.rtx`) and where
`\@extension` is the file extension for `\class`. For instance, to read in the file
`aps.rtx`, do `\class@extensionfile {aps} \substyle@ext`, where the latter has
been define to expand to `.rtx`.

Features supported include passing existing class options on to the class extension, `\AtEndOfClass` processing, a stack that restores `\@currname`, `\@currentx`, `\@clsextension`, and the `\catcode` of ‘@’, fall-back to a control sequence name (with leading ‘`rtx@`’) if no file exists.

Note that `\LoadClass` gives one the ability to write a class that calls in another class as a (sort of) module: this scheme is like `\LoadClass`, but turned inside out.

```

398 \def\class@extension#1#2{%
399 \IfFileExists{#1.#2}{%
400 \expandafter\class@extensionfile\csname ver@\@currname.\@currentx\endcsname{#1}#2%
401 }{%
402 \csname rtx@#1\endcsname
403 }%
404 }%
405 \def\class@extensionfile#1#2#3{%
406 \@pass@options#3\@unusedoptionlist{#2}%
407 \global\let\@unusedoptionlist\@empty
408 \expandafter\class@ext@hook\csname#2.#3-h@k\endcsname#1{#2}#3%
409 }%
410 \def\class@ext@hook#1#2#3#4{%
411 \@pushfilename@ltx
412 \makeatletter
413 \let\CurrentOption\@empty
414 \@reset@options
415 \let#1\@empty
416 \xdef\@currname{#3}%
417 \global\let\@currentx#4%
418 \global\let\@clsextension\@currentx
419 \input{#3.#4}%
420 \@ifl@ter#4{#3}#2{%
421 \class@info{Class extension later than: #2}%
422 }{%
423 \class@info{Class extension earlier: #2}%
424 \@@end
425 }%
426 #1%

```

```

427 \let#1\@undefined
428 \expandafter\p@pfilename@ltx\@currnamestack@ltx\@nil
429 \@reset@options
430 }%

```

`\@pushfilename` But! L^AT_EX does not provide for a class extension other than `.cls`, therefore we must extend L^AT_EX's file name stack with the file extension of a class extension. This way, procedures like `\ProvidesPackage`, `\OptionNotUsed`, `\ProcessOptions`, `\@reset@options` will still work properly.

```

431 \def\@pushfilename@ltx{%
432 \xdef\@currnamestack@ltx{%
433   {\@currname}%
434   {\@currentx}%
435   {\@clsextension}%
436   {\the\catcode'\@}%
437   \@currnamestack@ltx
438 }%
439 }%
440 \def\p@pfilename@ltx#1#2#3#4#5\@nil{%
441 \gdef\@currname{#1}%
442 \gdef\@currentx{#2}%
443 \gdef\@clsextension{#3}%
444 \catcode'\@#4\relax
445 \gdef\@currnamestack@ltx{#5}%
446 }%
447 \global\let\@currnamestack@ltx\@empty

```

We carefully patch L^AT_EX so that the current value of `\@clsextension` can be restored after reading in a class file.

6.8 Type Tools

`\flushing` Undoes `\centering`. Should also undo `\raggedleft` and `\raggedright`.

```

448 \def\flushing{%
449 \let\\\@normalcr
450 \leftskip\z@skip
451 \rightskip\z@skip
452 \@rightskip\z@skip
453 \parfillskip\@flushglue
454 }%

```

`\@centercr` The `\@centercr` command is the replacement for `\@normalcr` when setting type centered or ragged. Normally, the meaning of `\\` is `\@normalcr`, which L^AT_EX defines via `\DeclareRobustCommand`. In centered or ragged typesetting, the meaning of `\\` is `\@centercr`, therefore it ought to be defined via `\DeclareRobustCommand` (but unfortunately is not). The fact that it is not is yet another of L^AT_EX's early failures that will never get fixed.

The following exemplar fails under L^AT_EX version 2005/12/01, package `textcase` 2004/10/07 v0.07:

```
%\documentclass{article}%
%\usepackage[overload]{textcase}
%\begin{document}
%\centering
%\section{\MakeTextUppercase{Section\\title}}
%Text
%\end{document}
%
```

The solution is to promote `\@centercr` to a robust command, just the same as `\.`. We do that here without needing to know the meaning of the command.

```
455 \expandafter\DeclareRobustCommand\expandafter\@centercr\expandafter{\@centercr}%
```

6.9 Display Math

`\eqnarray@LaTeX` Team L^AT_EX has stated they will never repair Leslie's broken definition of `\eqnarray`. Let us be bold...

Note on `hyperref` package compatibility: that package overrides `\eqnarray` by wrapping it up in a larger procedure, so its changes are compatible with this package's changes.

```
456 \def\eqnarray@LaTeX{%
457   \stepcounter{equation}%
458   \def\@currentlabel{\p@equation\theequation}%
459   \global\@eqnswtrue
460   \m@th
461   \global\@eqcnt\z@
462   \tabskip\@centering
463   \let\@@@eqnocr
464   $$\everycr{}\halign to\displaywidth\bgroup
465     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
466     &\global\@eqcnt\@ne\hskip \tw@\arraycolsep \hfil${##}$\hfil
467     &\global\@eqcnt\tw@ \hskip \tw@\arraycolsep
468     $\displaystyle{##}$\hfil\tabskip\@centering
469     &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
470     \tabskip\z@skip
471   \cr
472 }
473 \long\def\eqnarray@fleqn@fixed{%
474   \stepcounter{equation}\def\@currentlabel{\p@equation\theequation}%
475   \global\@eqnswtrue\m@th\global\@eqcnt\z@
476   \tabskip\ltx@mathindent
477   \let\@=\@eqnocr
478   \setlength\abovedisplayskip{\topsep}%
479   \ifvmode\addtolength\abovedisplayskip{\partopsep}\fi
480   \addtolength\abovedisplayskip{\parskip}%
481   \setlength\belowdisplayskip{\abovedisplayskip}%
```

```

482 \setlength\belowdisplayshortskip{\abovedisplayskip}%
483 \setlength\abovedisplayshortskip{\abovedisplayskip}%
484 $$%
485 \everycr{}%
486 \halign@\linewidth\bgroup
487 \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
488 &\global\@eqcnt\@ne
489 \hskip\tw\@eqncolsep
490 \hfil${}\##{}}$\hfil
491 &\global\@eqcnt\tw@
492 \hskip\tw\@eqncolsep
493 $\displaystyle{##}$\hfil\tabskip\@centering
494 &\global\@eqcnt\thr@\hb@xt@\z@\bgroup\hss##\egroup
495 \tabskip\z@skip
496 \cr
497 }%
498 \@ifx{\eqnarray\eqnarray@LaTeX}{-%
499 \class@info{Repairing broken LaTeX eqnarray}%
500 \let\eqnarray\eqnarray@fleqn@fixed
501 \newlength\eqncolsep
502 \setlength\eqncolsep\z@
503 \let\eqnarray@LaTeX\relax
504 \let\eqnarray@fleqn@fixed\relax
505 }{}%

```

The macro `\ltx@mathindent` is assigned to the `\tabskip` glue just before the alignment preamble is expanded, the value therefore applying at the left of the first column.

The below value specifies the display math to be set centered, as is common practice. Alternatively, `\tabskip` can be set to a different glue value, accomplishing flush-left display math.

Note that the `ltxutil.dtxfleqn.clo` package provides its own meaning for the `eqnarray` environment, which is also broken. We do not patch that package, however.

Bug note: The `ltxutil.dtxlineno.sty` package detects `ltxutil.dtxfleqn.clo` by testing whether `\mathindent` is defined, instead of using correct L^AT_EX 2_ε means. Even though our `eqnarray` environment is modelled after `ltxutil.dtxfleqn.clo`, we must program defensively here.

```

506 \def\ltx@mathindent{\@centering}%
507 \def\set@eqnarray@skips{}%

```

`\prep@math` If we are in vertical mode when display math mode is entered (via `$$`), T_EX will first enter horizontal mode, then display math mode; this results in a phantom paragraph containing a single `\hbox` consisting of the `\parindent` box followed by the `\parskipfillskip` glue. Of course, that `\hbox` is accompanied by `\parskip` glue and `\baselineskip` glue.

The `\prep@math` procedure removes the `\parindent` box, thereby (magically) eliminating the phantom paragraph. The `\prep@math@patch` procedure head-

patches the `equation` and `eqnarray` environments to accomplish this removal of the phantom paragraph.

Note that there are three remaining ways to enter display math mode that we do not treat: the `displaymath` environment (equivalent to `\[/\]`), and the primitive the `$$` markup. I refrain from treating the first case because `displaymath` already detects the case where it is entered from vertical mode: I do not wish to engage in the dubious enterprise of attempting to correct a procedure that is ill conceived from the outset. As to the primitive `$$`, there is no help for users who insist upon employing procedural markup in their documents. in their documents.

```
508 \def\prep@math{%
509 \@ifvmode{\everypar{\setbox\z@\lastbox}}{}}%
510 }%
511 \def\prep@math@patch{%
512 \prepdef\equation{\prep@math}%
513 \prepdef\eqnarray{\prep@math}%
514 }%
```

A document class may invoke `\prep@math@patch` at any point it wishes to prevent the appearance of the phantom paragraph: it may be a global declaration or a local one.

We fail to patch `\[, \equation`, however.

6.10 Footnotes

<code>\footnotemark</code>	We repair an error in the L ^A T _E X kernel (see <code>lfloat.dtx</code>) involving footnotes.
<code>\footnotetest</code>	The symptom is that the <code>\footnotemark</code> command does not work properly
<code>\ltx@xfootnote</code>	within a <code>minipage</code> environment. The source of the problem is in the way the
<code>\ltx@footmark</code>	<code>\footnotemark</code> and <code>\@xfootnotemark</code> procedures are defined: they do not share
<code>\ltx@foottext</code>	the method, used by <code>\footnote</code> and other procedures, that allows a context switch
<code>\ltx@make@current@footnote</code>	to change the way footnotes behave within a <code>minipage</code> environment. This is a

L^AT_EX bug of long standing; our fix dates to 1987.

While we are at it, we rewrite both the `\footnote`, `\footnotemark` and `\footnotetest` procedures, achieving a cleaner separation of syntax and semantics. Note that the `\@footnotetest` procedure is not involved in context switching; `hyperref` will take over that procedure, substituting its own processing around its argument and passing this to `\H@@footnotetest`. We anticipate this, and do our context switching on `\H@@footnotetest`.

The `\@makefnmark` continues as the method of formatting the footnote mark.

A note about the context switch mentioned above: the `minipage` environment executes the following in order to alter the way footnotes behave:

```
%\def\@mpfn{mpfootnote}%
%\def\thempfn{\thempfootnote}%
%\let\@footnotetest\@mpfootnotetest
%\let\@makefnmark\@mpmakefnmark
%\c@mpfootnote\z@
```

This code changes the counter used in autonumbered footnotes, the choice of footnote marker, and the procedure used on the footnote text. Changing the counter is needed because minipage footnotes are in their own sequence, and the footnote marker is customarily different within a minipage. The procedure that works on the footnote text must be different because the footnotes are placed at the bottom of the minipage, not the bottom of the text column.

Note that L^AT_EX initially defines `\@mpfn` as `footnote` and `\thempfn` as `\thefootnote`, so we are initially doing general footnotes.

Any procedure that establishes a minipage-like context (e.g., floats) can do the same as the minipage context switch illustrated above.

Three user-level command, `\footnote`, `\footnotemark`, and `\footnotetext` are defined (see the L^AT_EX manual for user-level details).

`\footnote` The first user-level command is `\footnote`. A simple way to look at this command is to think of it as `\footnotemark [number] \footnotetext [number] {text}`, where the optional argument is the same in both calls. We also define a syntactical helper procedure `\ltx@xfootnote`.

We employ the procedures `\ltx@stp@footproc` and `\ltx@def@footproc`, passing in the procedure to execute, in this case `\ltx@footmark`, which sets the footnote mark. In any case, we end on the procedure `\ltx@foottext`, which sets the footnote text.

```
515 \def\footnote{\@ifnextchar[\ltx@xfootnote\ltx@yfootnote}%
516 \def\ltx@xfootnote[#1]{%
517 \ltx@def@footproc\ltx@footmark[#1]%
518 \expandafter\ltx@foottext\expandafter{\the\csname c@\@mpfn\endcsname}%
519 }%
520 \def\ltx@yfootnote{%
521 \ltx@stp@footproc\ltx@footmark
522 \expandafter\ltx@foottext\expandafter{\the\csname c@\@mpfn\endcsname}%
523 }%
```

The `\footmark` user-level command is next. Here we use the procedures `\ltx@stp@footproc` and `\ltx@def@footproc` again, but unlike `\footnote`, we do not set the footnote text.

```
524 \def\footnotemark{\@ifnextchar[\ltx@xfootmark\ltx@yfootmark}%
525 \def\ltx@xfootmark{\ltx@def@footproc\ltx@footmark}%
526 \def\ltx@yfootmark{\ltx@stp@footproc\ltx@footmark}%
527 \def\ltx@footmark#1{%
528 \leavevmode
529 \ifhmode\edef\@x@sf{\the\spacefactor}\nobreak\fi
530 \begingroup
531 \expandafter\ltx@make@current@footnote\expandafter{\@mpfn}{#1}%
532 \expandafter\@argswap@val\expandafter{\Hy@footnote@currentHref}{\hyper@linkstart {link}}%
533 \@makefnmark
534 \hyper@linkend
535 \endgroup
536 \ifhmode\spacefactor\@x@sf\fi
537 \relax
538 }%
```

The third user-level command is `\footnotetext`. As with `\footnotemark`, we use the procedures `\ltx@stp@footproc` and `\ltx@def@footproc`, this time passing in the procedure `\ltx@foottext`, which sets the footnote text.

```

539 \def\footnotetext{\@ifnextchar[\ltx@xfoottext\ltx@yfoottext]}%
540 \def\ltx@xfoottext{\ltx@def@footproc\ltx@foottext}%
541 \def\ltx@yfoottext{\ltx@stp@footproc\ltx@foottext}%
542 \long\def\ltx@foottext#1#2{%
543   \begingroup
544   \expandafter\ltx@make@current@footnote\expandafter{\@mpfn}{#1}%
545   \@footnotetext{#2}%
546   \endgroup
547 }%

```

Here are the definitions of the procedures `\ltx@stp@footproc` and `\ltx@def@footproc`. The `require` argument is the procedure to execute afterwards, and `\ltx@def@footproc` parses a bracket-delimited argument (it is not optional). In each case the given procedure is executed with an argument prepared for it: the value of the footnote counter.

```

548 \def\ltx@def@footproc#1[#2]{%
549   \begingroup
550   \csname c@\@mpfn\endcsname #2\relax
551   \unrestored@protected@xdef\@thefnmark{\thempfn}%
552   \expandafter\endgroup
553   \expandafter#1%
554   \expandafter{\the\csname c@\@mpfn\endcsname}%
555 }%
556 \def\ltx@stp@footproc#1{%
557   \expandafter\stepcounter\expandafter{\@mpfn}%
558   \protected@xdef\@thefnmark{\thempfn}%
559   \expandafter#1%
560   \expandafter{\the\csname c@\@mpfn\endcsname}%
561 }%

```

Here we provide for our good friend `hyperref` to enter in like a bull in a china shop. If it is not loaded, we do what it would have done, but gentler and without hypertext functionality.

```

562 \appdef\class@documenthook{%
563   \let\footnote@latex\footnote
564   \@ifpackageloaded{hyperref}{-}{%
565     \let\H@@footnotetext\@footnotetext
566     \def\@footnotetext{\H@@footnotetext}%
567     \let\H@@mpfootnotetext\@mpfootnotetext
568     \def\@mpfootnotetext{\H@@mpfootnotetext}%
569   }%
570 }%

```

In the following, we must use L^AT_EX's rococco equipment in the form of `\protected@edef`, because of the presence of a font switch in the meaning of `\thempfootnote`. But, really, isn't this a sloppy conflation of semantics and presentation?


```

571 \def\ltx@make@current@footnote#1#2{%
572   \csname c@#1\endcsname#2\relax
573   \protected@edef\Hy@footnote@currentHref{\@currentHref-#1.\csname the#1\endcsname}%
574 }%
575 \def\thempfootnote@latex{{\itshape \@alph \c@mpfootnote }}%
576 \def\ltx@thempfootnote{\@alph\c@mpfootnote}%
577 \@ifx{\thempfootnote\thempfootnote@latex}{%
578   \class@info{Repairing hyperref-unfriendly LaTeX definition of \string\mpfootnote}%
579   \let\thempfootnote\ltx@thempfootnote
580 }{}%

```

Note on `hyperref` compatibility: In its “Automated L^AT_EX hypertext cross-references”, the `hyperref` package alters footnote processing, but it does nothing to address the several issues of concern to us.

The `hyperref` package takes over the `\@mpfootnotetext` and `\@footnotetext` procedures, wrapping the argument in its own code. It also rewrites `\@footnotemark`, making it a hyperlink, and `\@xfootnotenext`, removing from it all hypertext capabilities.

However, if the `\footnotemark` command has been supplied with an optional argument, `hyperref`’s changes do not apply: it punts in this case.

At the same time, it attempts to turn off its changes during `\maketitle` processing, destroying one of the capabilities we desire.

We make ourself `hyperref` savvy: we re-implement footnote processing, using `hyperref` capabilities if that package has been loaded.

Any other package that rewrites L^AT_EX’s footnote macros will be incompatible with this package.

Two thoughts about `hyperref`: what for does it define `\realfootnote`? Apparently even SR himself cannot remember.

Also: a document class that desires high hypertext capabilities might well wish to reimplement `\maketitle` so that footnotes called out from there are hypertext links: the `hyperref` package’s “Automated L^AT_EX hypertext cross-references” does not do any of this:

But the special footnotes in `\maketitle` are much too hard to deal with properly. Let them revert to plain behaviour.

Note that the document class, in reimplementing `\maketitle`, must ensure that the `hyperref` package does not clobber its own definition!

<code>\@footnotetext</code>	The two procedures <code>\@footnotetext</code> and <code>\@mpfootnotetext</code> share code. We
<code>\@mpfootnotetext</code>	make that explicit here.
<code>\@tpfootnotetext</code>	Note that the procedure calling <code>\make@footnotetext</code> will open a group with
<code>\make@footnotetext</code>	<code>\bgroup</code> which is then closed by <code>\minipagefootnote@drop</code> .
<code>\set@footnotewidth</code>	Difference from L ^A T _E X: here we do not set <code>\floatingpenalty</code> to infinity. Doing
	this must date back to a time when L ^A T _E X could not accomodate split insertions
	(footnotes). I cannot think of any other reason to do have done this. At any rate,
	with the <code>ltxgrid</code> package, split insertions are properly taken care of, so we allow
	it.

We provide the hook `\set@footnotewidth` that sets the footnote on a particular measure. Some page grids are such as to set a footnote in a context where `\columnwidth` is not the right parameter to use for the set width of a footnote. In such a case, for the applicable scope, you should define `\set@footnotewidth` to perform this job correctly.

If we are setting type on multiple page grids, we must still ensure that all footnotes that find their way into the `\footins` insert register are set on the same width. This implies the need for a document to have an “overall” page grid, which determines the set width of all footnotes with the exception of minipage footnotes.

In general, remember that footnotes, like all insertions (including floats), are a step outside of the galley context, and all aspects of insertions need to be properly handled, including the set width.

```

581 \def\@makefnmark{%
582   \hbox{%
583     \@textsuperscript{%
584       \normalfont\itshape\@thefnmark
585     }%
586   }%
587 }%

588 \long\def\@footnotetext{%
589   \insert\footins\bgroup
590   \make@footnotetext
591 }%

592 \long\def\@mpfootnotetext{%
593   \minipagefootnote@pick
594   \make@footnotetext
595 }%

```

Procedure `\make@footnotetext` sets the footnote #1 into type, with the proper font, color, leading, width, and label in effect. It also establishes a strut and null glue at the end of the last paragraph of the footnote; The strut helps compensate for the lack of `\interlineskip` glue between `\inserts`; the glue establishes a feasible `\vsplit` point between footnotes.

Note that in the title block (`ltxfront`), the alternative definition, under the name `\frontmatter@footnotetext`, is used. The only material difference there is the reference to `\frontmatter@makefnntext` instead of `\@makefnntext`.

Dependency note: the `\@makefnntext` procedure is used to further process the footnote text and to execute the `\@makefnmark` procedure to produce the footnote mark. The definition of the former is customarily found in the document class (hereunder that of `ltxutil.dtxarticle.cls`), the latter in `ltxutil.dtxlatex.ltx`. They are as follows:

```

%\newcommand\@makefnntext[1]{%
% \parindent 1em\noindent
% \hb@xt@1.8em{\hss\@makefnmark}%
% #1%
%}%

```

```

%\def\@makefnmark{%
% \hbox{\@textsuperscript{\normalfont\@thefnmark}}}%
}%%
%
```

```

596 \long\def\make@footnotetext#1{%
597 \set@footnotefont
```

As noted above, we do *not* do `\floatingpenalty \@MM`, as in standard L^AT_EX.

```

598 \set@footnotewidth
599 \@parboxrestore
600 \protected@edef\@currentlabel{%
```

Note that we employ `\@mpfn` as a level of redirection for the footnotecounter.

```

601 \csname p@\@mpfn\endcsname\@thefnmark
602 }%
603 \color@begingroup
604 \@makefntext{%
605 \rule\z@\footnotesep\ignorespaces#1%
```

The following strut and glue are for spacing and splitting, as mentioned above.

```

606 \@finalstrut\strutbox\vadjust{\vskip\z@skip}%
607 }%
608 \color@endgroup
609 \minipagefootnote@drop
610 }%
```

`\set@footnotefont` is the procedure for setting the font of a footnote. Other aspects of the environment may be set using this hook.

```

611 \def\set@footnotefont{%
612 \reset@font\footnotesize
613 \interlinepenalty\interfootnotelinepenalty
614 \splittopskip\footnotesep
615 \splitmaxdepth\dp\strutbox
616 }%
```

`\set@footnotewidth` is the procedure for setting the width of a footnote. The default page grid, a single, full-width column, sets footnotes on the width of the text.

```

617 \def\set@footnotewidth{\set@footnotewidth@one}%
```

6.11 Floats

6.11.1 Usage notes

We extend the L^AT_EX kernel for three purposes:

1. When the `\footnote` command is used within the scope of a float, we do as `minipage` does.
2. We provide a mechanism to write floats out to an external stream for temporary storage (deferred floats).

3. We provide mechanism for placing a float **here** invariably, that is, floats are unfloat. This mechanism is used to read the external stream mentioned above.

To use these mechanisms, the document class should define a float, say, **figure** as per usual, and in addition:

1. Optionally define an alternative, say **figure@write** as follows:

```
\newenvironment{figure@write}{%
% \write@float{figure}%
%}{%
% \endwrite@float
%}
```

That is, the alternative environment executes `\write@float` instead of `\@float`. Note that this step is not needed if the float environment is defined in the simple way of `classes.dtx`. However, an environment like `longtable` will require it.

2. Install into `\AtBeginDocument` a call to `\do@if@floats`, with the float name and an appropriate file extension as its arguments.

```
\appdef\class@documenthook{\do@if@floats{figure}{.fgx}}
```

3. Optionally define a text entity `\figuresname` that will be the text of the head that is set over the deferred floats. If not defined, there will be no head.
4. Optionally define a user-level command to allow the document to determine where the figures are printed out (default is to print at end of document). E.g.,

```
\newcommand\printfigures{\print@float{figure}}
```

5. Install into `\appdef\class@enddocumenthook` a call to `\printfigures`, or, if the latter is not defined, as follows:

```
\appdef\class@enddocumenthook{\print@float{figure}}
```

Note that installing this command into `\AtBeginDocument` is best done earlier than calls that assume the last page of the document is at hand.

6.11.2 Robustifying fragile commands

Certain of L^AT_EX's commands cannot be written out to a file or appear within a `\mark` command argument because they do calculations during expansion. We provide for a little help, but without changing the meanings of these commands.

```
\addtocontents
\robustify@contents 618 \def\robustify@contents{%
                    619 \let \label \gobble
                    620 \let \index \gobble
                    621 \let \glossary \gobble
                    622 \let\footnote \gobble
                    623 \def\({\string\}%
                    624 \def\){\string\})%
                    625 \def\\{\string\\}%
                    626 }%
                    627 \long\def\addtocontents#1#2{%
                    628 \protected@write\@auxout{\robustify@contents}{\string \@writefile {#1}{#2}}%
                    629 }%
```

6.11.3 Preparing for the hyperref package

```
\addcontentsline \label \ltx@contentsline
```

The `hyperref` package assumes that the `\contentsline` command will be given four arguments. Therefore it cannot successfully process a `ltxutil.dtx.toc` file that had been written by standard L^AT_EX. We fix things up by always writing that fourth argument and by supplying a `\contentsline` command that can read them.

We also give the `\newlabel` command's second argument five tokens.

Finally, we wrap L^AT_EX's `\contentsline` command with code to detect the case where the expected procedure is not defined, and we give it a syntax with no semantics.

We switch over to this new definition only after `hyperref` has loaded.

```
630 \def\addcontentsline#1#2#3{%
631 \addtocontents{#1}{%
632 \protect\contentsline{#2}{#3}{\thepage}{}%
633 }%
634 }%
635 \def\label#1{%
636 \@bsphack
637 \protected@write\@auxout{%
638 \string\newlabel{#1}{\@currentlabel}{\thepage}{}}}%
639 }%
640 \@esphack
641 }%
642 \def\ltx@contentsline#1{%
643 \expandafter\@ifnotrelax\csname l@#1\endcsname}{}%
644 \expandafter\let\csname l@#1\endcsname@gobbletwo
645 }%
646 \contentsline@latex{#1}%
647 }%
```

```

648 \appdef\document@inithook{%
649 \let\contentsline@latex\contentsline
650 \let\contentsline\ltx@contentsline
651 }%

```

6.11.4 Footnotes within floats, unfloating floats, float font

`\caption` DPC: Er a bit of a hack, but seems best way of supporting normal L^AT_EX syntax at this point: If a caption is used below a table, then put out the footnotes before the caption.

```

652 \appdef\class@documenthook{%
653 \predef\caption{\minipagefootnote@here}%
654 }%

```

Note on `hyperref` compatibility: this change to the `\caption` command is compatible with the “Automated L^AT_EX hypertext cross-references” patches of that package.

All the same, I think Sebastian’s changes to `\caption` and `\@caption` could bear with some improvement. The following implementation requires knowing only the pattern part of the `\@caption` macro:

```

%\def\caption{%
% \H@refstepcounter\@capttype
% \hyper@makecurrent{\@capttype}%
% \@dblarg{\H@caption\@capttype}%
%}%
%\def\H@caption#1[#2]#3{%
% \@caption{#1}[#2]{%
% \ifHy@nesting
% \hyper@@anchor{\@currentHref}{#3}%
% \else
% \hyper@@anchor{\@currentHref}{\relax}#3%
% \fi
% }%
%}

```

`\minipagefootnote@init` Procedure to deal with footnotes accumulated within a minipage environment.

`\minipagefootnote@here` These procedures encapsulate all uses of the `\@mpfootins` box.

`\minipagefootnote@foot` Note: `\minipagefootnote@here` must *not* be executed within the MVL!

```

\minipagefootnote@pick 655 \def\minipagefootnote@init{%
\minipagefootnote@drop 656 \setbox\@mpfootins\box\voidb@x
657 }%
658 \def\minipagefootnote@pick{%
659 \global\setbox\@mpfootins\ vbox\bgroup
660 \unvbox\@mpfootins
661 }%
662 \def\minipagefootnote@drop{%
663 \egroup
664 }%

```

```

665 \def\minipagefootnote@here{%
666   \par
667   \@ifvoid\@mpfootins{}{%
668     \vskip\skip\@mpfootins
669     \fullinterlineskip
670     \@ifinner{%
671       \vtop{\unvcopy\@mpfootins}%
672       {\setbox\z@\lastbox}%
673     }{%
674       \unvbox\@mpfootins
675     }%
676 }%
677 \def\minipagefootnote@foot{%
678   \@ifvoid\@mpfootins{}{%
679     \insert\footins\bgroup\unvbox\@mpfootins\egroup
680   }%
681 }%
682 \def\endminipage{%
683   \par
684   \unskip
685   \minipagefootnote@here
686   \@minipagefalse %% added 24 May 89
687   \color@endgroup
688   \egroup
689   \expandafter\@iiiparbox\@mpargs{\unvbox\@tempboxa}%
690 }%

```

`\floats@sw` The Boolean `\floats@sw` signifies that floats are to be floated; if false, that floats are to be deferred to the end of the document. Note that the assignment of this Boolean is to be overridden by the document class in response to user-selected options.

```
691 \@booleantrue\floats@sw
```

`\@xfloat` The float start-code is redefined to set up footnotes in the style of minipage. Also, `\@mpmakefn` the `\floats@sw` Boolean informs us that floats are to be all placed **here**. Note that, to protect against the Boolean being undefined at this late hour, we default it globally to true.

```

692 \let\@xfloat@LaTeX\@xfloat
693 \def\@xfloat#1[#2]{%
694   \@xfloat@prep
695   \@nameuse{fp@proc#2}%
696   \floats@sw{\@xfloat@LaTeX{#1}[#2]}{\@xfloat@anchored{#1}[]}%
697 }%
698 \def\@xfloat@prep{%
699   \ltx@footnote@pop
700   \def\@mpfn{mpfootnote}%
701   \def\thempfn{\thempfootnote}%
702   \c@mpfootnote\z@
703   \let\H@footnotetext\H@mpfootnotetext

```

```

704 }%
705 \let\ltx@footnote@pop\@empty
706 \def\xfloat@anchored#1[#2]{%
707 \def\@captype{#1}%
708 \begin@float@pagebreak
709 \let\end@float\end@float@anchored
710 \let\end@dblfloat\end@float@anchored
711 \hspace\columnwidth
712 \parboxrestore
713 \floatboxreset
714 \minipagefootnote@init
715 }%
716 \def\end@float@anchored{%
717 \minipagefootnote@here
718 \par\vskip\z@skip
719 \par
720 \end@float@pagebreak
721 }%
722 \def\begin@float@pagebreak{\par\addvspace\intextsep}%
723 \def\end@float@pagebreak{\par\addvspace\intextsep}%
724 \def\@mpmakeintext#1{%
725 \parindent=1em
726 \noindent
727 \hb@xt@1em{\hss\@makefnmark}%
728 #1%
729 }%

```

6.11.5 Writing floats out to a file

`\do@if@floats` The procedure `\do@if@floats` should be executed at `\class@documenthook` time: it arranges to write out the floats of the given class to a temporary file, to be read back later (deferred floats), given that `\floats@sw` is false. Note that, to protect against the Boolean being undefined at this late hour, we default it globally to true.

```

730 \def\do@if@floats#1#2{%
731 \floats@sw}{%

```

Open the stream to save out the document's floats of this class.

```

732 \expandafter\newwrite
733 \csname#1write\endcsname
734 \expandafter\def
735 \csname#1@stream\endcsname{\jobname#2}%
736 \expandafter\immediate
737 \expandafter\openout
738 \csname#1write\endcsname
739 \csname#1@stream\endcsname\relax

```

Swap environments. If the class writer has defined, e.g., `figure@write`, then we use this as the procedure to execute for writing the float out to the external

stream. Otherwise, the replacement of `\@float` by `\write@float` should do the right thing for float environments defined in the simple way of `classes.dtx`.

```

740 \@ifxundefined\@float@LaTeX{%
741   \let\@float@LaTeX\@float
742   \let\@dblfloat@LaTeX\@dblfloat
743   \let\@float\write@float
744   \let\@dblfloat\write@floats
745 }{%
746 \let@environment{#1@float}{#1}%
747 \let@environment{#1@floats}{#1*}%
748 \@ifxundefined@cs{#1@write}{-}{%
749   \let@environment{#1}{#1@write}%
750 }%
751 }%
752 }%

```

`\print@float` The procedure `\print@float` prints out the deferred floats.

Here, we make use of the `\floats@sw` Boolean to select the non-floating type of processing.

```

753 \def\triggerpar{\leavevmode\@par}%
754 \def\onepage{\def\begin@float@pagebreak{\newpage}\def\end@float@pagebreak{\newpage}}%
755 \def\print@float#1#2{%
756   \lengthcheck@sw{%
757     \total@float{#1}%
758   }{%
759     \@ifxundefined@cs{#1@write}{-}{%
760       \begingroup
761         \@booleanfalse\floats@sw
762         #2%
763         \raggedbottom
764         \def\array@default{v}% floats must
765         \let\@float\@float@LaTeX
766         \let\@dblfloat\@dblfloat@LaTeX
767         \let\trigger@float@par\triggerpar
768         \let@environment{#1}{#1@float}%
769         \let@environment{#1*}{#1@floats}%
770         \expandafter\prepdef\csname#1\endcsname{\trigger@float@par}%
771         \expandafter\prepdef\csname#1*\endcsname{\trigger@float@par}%
772         \@namedef{fps#1}{h!}%
773         \expandafter\immediate
774         \expandafter\closeout
775         \csname#1@write\endcsname
776         \everypar{%
777           \global\let\trigger@float@par\relax
778           \global\everypar{}\setbox\z@\lastbox
779           \@ifxundefined@cs{#1@name}{-}{%
780             \begin@float@pagebreak
781             \expandafter\section
782             \expandafter*%

```

```

783     \expandafter{%
784         \csname#1sname\endcsname
785     }%
786 }%
787 }%
788 \input{\csname#1@stream\endcsname}%
789 \endgroup
790 \global\expandafter\let\csname#1write\endcsname\relax
791 }%
792 }%

```

`\tally@float` If we are tallying column inches, `\tally@float` tallies a contribution to `\ftype@`
`\total@float` `\@capytype`, depending upon the width of `\@currbox`. In effect, each float class is tallied in two sections, one for narrow, one for wide floats.

If statistics are wanted, `\total@float` logs the tally for the given float class. The quantity `\@twopowerfourteen` is 2^{14} , `\@twopowertwo` is 2^2 .

```

793 \chardef\@xvi=16\relax
794 \mathchardef\@twopowerfourteen="4000
795 \mathchardef\@twopowertwo="4
796 \def\tally@float#1{%
797 \begingroup

```

We strip all but the least significant 5 bits from `\count \@currbox`, and put them into `\@tempcnta`. We then subtract 16 from `\count \@currbox` (unless this would make it negative), effectively reversing the process carried out in `\@float`.

```

798 \@tempcnta\count\@currbox
799 \divide\@tempcnta\@xxxii
800 \multiply\@tempcnta\@xxxii
801 \advance\count\@currbox-\@tempcnta
802 \divide\@tempcnta\@xxxii
803 \ifnum\count\@currbox>\@xvi}{%
804 \advance\count\@currbox-\@xvi\@booleantrue\@temp@sw
805 }{%
806 \@booleanfalse\@temp@sw
807 }%

```

If so desired, we log the characteristics of this float object: float class and float placement parameters, height, depth, and width.

```

808 \show@box@size@sw{%
809 \class@info{Float #1
810 (\the\@tempcnta)[\@temp@sw{16+}]{\the\count\@currbox}^^J%
811 (\the\ht\@currbox+\the\dp\@currbox)X\the\wd\@currbox
812 }%
813 }{}%
814 \endgroup

```

Here we tally the height of this float object.

```

815 \expandafter\let
816 \expandafter\@tempa
817 \csname fbox@\csname ftype@#1\endcsname\endcsname

```

```

818 \ifnotrelax\@tempa{%
819 \@ifhbox\@tempa{%
820 \setbox\@tempboxa\vbox{\unvcopy\@currbox\hrule}%
821 \dimen@ht\@tempboxa
822 \divide\dimen@\@twopowerfourteen
823 \@ifdim{\wd\@tempboxa<\textwidth}{%
824 \advance\dimen@ht\@tempa
825 \global\ht\@tempa\dimen@
826 }{%
827 \advance\dimen@dp\@tempa
828 \global\dp\@tempa\dimen@
829 }%
830 }{}%
831 }{}%
832 }%
833 \def\total@float#1{%
834 \expandafter\let
835 \expandafter\@tempa
836 \csname fbox@\csname ftype#1\endcsname\endcsname
837 \ifnotrelax\@tempa{%
838 \@ifhbox\@tempa{%
839 \@tempdima\theht\@tempa\divide\@tempdima\@twopowertwo\@tempcnta\@tempdima
840 \@tempdimb\thedp\@tempa\divide\@tempdimb\@twopowertwo\@tempcntb\@tempdimb
841 \class@info{Total #1: Column(\the\@tempcnta pt), Page(\the\@tempcnta pt)}}%
842 }{}%
843 }{}%
844 }%

```

`\write@float` Handles the case where the name of the float is the same as that of the stream.
`\write@floats` Note that `longtable` does *not* fit this case. Note also: `\write@float` is *not* a
`\write@@float` user-level environment, therefore it is properly not defined with `\newenvironment`.

```

845 \def\write@float#1{\write@@float{#1}{#1}}%
846 \def\endwrite@float{\@Esphack}%
847 \def\write@floats#1{\write@@float{#1*}{#1}}%
848 \def\endwrite@floats{\@Esphack}%

```

`\write@@float`

```

849 \def\write@@float#1#2{%
850 \ifhmode
851 \@bsphack
852 \fi
853 \chardef\@tempc\csname#2write\endcsname
854 \toks@\{\begin{#1}\}%
855 \def\@tempb{#1}%
856 \expandafter\let\csname end#1\endcsname\endwrite@float
857 \catcode'\^M\active
858 \@makeother\{\@makeother\}\@makeother\%
859 \write@floatline
860 }%

```

```

\write@floatline The procedure \write@floatline only parses; it passes its result to \@write@floatline,
\@write@floatline which writes the line to output, then tests the line for the \end{float} tokens
\float@end@tag with aid of the \float@end@tag procedure.
861 \begingroup
862 \catcode'\[\the\catcode'\{\catcode'\}\the\catcode'\}\@makeother\{\@makeother\}%
863 \gdef\float@end@tag#1\end{#2}#3\nul[%
864 \def\@tempa[#2]%
865 \@ifx[\@tempa\@tempb][\end{#2}][\write@floatline]%
866 ]%
867 \obeylines%
868 \gdef\write@floatline#1^M[%
869 \begingroup%
870 \newlinechar'\^M%
871 \toks@\expandafter[\the\toks@#1]\immediate\write\@tempc[\the\toks@]%
872 \endgroup%
873 \toks@[]%
874 \float@end@tag#1\end{ }\@nul%
875 ]%
876 \endgroup

```

6.12 Counters

The following definitions override those of the L^AT_EX kernel, providing for a greater range of inputs.

```

877 \def\@alph#1{\ifcase#1\or a\or b\or c\or d\else\@ialph{#1}\fi}
878 \def\@ialph#1{\ifcase#1\or \or \or \or \or e\or f\or g\or h\or i\or j\or
879 k\or l\or m\or n\or o\or p\or q\or r\or s\or t\or u\or v\or w\or x\or
880 y\or z\or aa\or bb\or cc\or dd\or ee\or ff\or gg\or hh\or ii\or jj\or
881 kk\or ll\or mm\or nn\or oo\or pp\or qq\or rr\or ss\or tt\or uu\or
882 vv\or ww\or xx\or yy\or zz\else\@ctrerr\fi}

```

6.13 Customization of Sections

Patch the standard L^AT_EX sectioning procedure to:

- Allow a sectioning command to trigger the title page, or more generally to recognize that it is the first object in the document, so we headpatch `\@startsection`.
- Allow a tail command in #6 to uppercase the title, so we retain DPC's braces.
- Allow each type of sectioning command to format its number differently, so we generalize `\@secntformat`.
- Allow each type of sectioning command to format its argument differently, so we generalize `\@hangfrom`.
- Allow the starred form of the command to mark (the running head) and make an entry in the TOC, so we put `\@ssect` on the same footing as `\@sect`.

Note that the tokens passed to the TOC now are *not* the optional argument of the command, but the required. This means that the user can no longer use the former to put variant content in to the TOC as the Manual says.

Instead, the optional argument is used to put an alternative title into the running headers, a better choice.

`\@startsection` Patch a head hook into the basic sectioning command. Treat `\@sect` and `\@ssect` on an equal footing; now their pattern parts are identical.

```

883 \def\@startsection#1#2#3#4#5#6{%
884 \@startsection@hook
885 \ifnoskipsec \leavevmode \fi
886 \par
887 \@tempskipa #4\relax
888 \@afterindenttrue
889 \ifdim \@tempskipa <\z@
890 \@tempskipa -\@tempskipa \@afterindentfalse
891 \fi
892 \if@nobreak
893 \everypar{}%
894 \else
895 \addpenalty\@secpenalty\advspace\@tempskipa
896 \fi
897 \@ifstar
898 {\@dblarg{\@ssect@ltx{#1}{#2}{#3}{#4}{#5}{#6}}}%
899 {\@dblarg{\@sect@ltx {#1}{#2}{#3}{#4}{#5}{#6}}}%
900 }%
901 \def\@startsection@hook{}%
```

`\@sect` When defining `\@svsec`, do not expand `\@secntformat`. Put brace characters back where they were before David Carlisle got at them (i.e., as if `\@hangfrom` had two arguments). Protect the mark mechanism from an undefined meaning. Pass #8 to the TOC instead of #7. Remove `\relax` from the replacement part of `\@svsec`.

The procedure `\@changfrom` and `\@runinto` can be used to process the argument of the head. The head can define, e.g., `\@changfrom@section`, to do its own processing.

In using `\H@refstepcounter` in place of `\refstepcounter` we rely on either loading before any package that patches the latter, or the convention that the former is the original L^AT_EX procedure.

```

902 \class@info{Repairing broken LaTeX \string\@sect}%
903 \def\@sect@ltx#1#2#3#4#5#6[#7]#8{%
904 \@ifnum{#2}>\c@secnumdepth}{%
905 \def\H@svsec{\phantomsection}%
906 \let\@svsec\@empty
907 }{%
908 \H@refstepcounter{#1}%
909 \def\H@svsec{%
910 \phantomsection
```

```

911 }%
912 \protected@edef\@svsec{#{1}}%
913 \@ifundefined{#1cntformat}{%
914 \prepdef\@svsec\@seccntformat
915 }{%
916 \expandafter\prepdef
917 \expandafter\@svsec
918 \csname #1cntformat\endcsname
919 }%
920 }%
921 \@tempskipa #5\relax
922 \@ifdim{\@tempskipa>\z@}{%
923 \begingroup
924 \interlinepenalty \@M
925 #6{%
926 \@ifundefined{@hangfrom#1}{\@hangfrom}{\csname @hangfrom#1\endcsname}%
927 {\hskip#3\relax\H@svsec}{\@svsec}{#8}%
928 }%
929 \@@par
930 \endgroup
931 \@ifundefined{#1mark}{\@gobble}{\csname #1mark\endcsname}{#7}%
932 \addcontentsline{toc}{#1}{%
933 \@ifnum{#2>\c@secnumdepth}{%
934 \protect\numberline{}}%
935 }{%
936 \protect\numberline{\csname the#1\endcsname}%
937 }%
938 #8}%
939 }{%
940 \def\@svsechd{%
941 #6{%
942 \@ifundefined{@runin@to#1}{\@runin@to}{\csname @runin@to#1\endcsname}%
943 {\hskip#3\relax\H@svsec}{\@svsec}{#8}%
944 }%
945 \@ifundefined{#1mark}{\@gobble}{\csname #1mark\endcsname}{#7}%
946 \addcontentsline{toc}{#1}{%
947 \@ifnum{#2>\c@secnumdepth}{%
948 \protect\numberline{}}%
949 }{%
950 \protect\numberline{\csname the#1\endcsname}%
951 }%
952 #8}%
953 }%
954 }%
955 \@xsect{#5}%
956 }%
957 \def\@hang@from#1#2#3{\@hangfrom{#1#2}#3}%
958 \def\@runin@to #1#2#3{#1#2#3}%

```

\@ssect Put brace characters back where they were before David Carlisle got at them (as

if `\@hangfrom` has two arguments). Possibly set a mark. Make a TOC entry.

Note that, for compatibility with the `hyperref` package, we need to provide the interface required by that package (actually required by `pdfmark.def` and `nameref.sty`), namely the definition of `\@currentlabelname` (but now removed), the insertion of the procedure `\Sectionformat` (but why is this needed?), and the call to `\phantomsection` (which must precede the call to `\addcontentsline`). We also have to sidestep the patch to `\@ssect` in that same file, therefore we use a different control sequence name in the call from `\@startsection`.

```

959 \def\@ssect@ltx#1#2#3#4#5#6[#7]#8{%
  Removed \def\@currentlabelname{#8}
960 \def\H@svsec{\phantomsection}%
961 \@tempskipa #5\relax
962 \@ifdim{\@tempskipa>\z@}{%
963   \begingroup
964     \interlinepenalty \@M
965     #6{%
966       \@ifundefined{@hangfroms@#1}{\@hang@froms}{\csname @hangfroms@#1\endcsname}%
  Removed {\hskip#3\relax\H@svsec}{\Sectionformat{#8}{#1}}
967     {\hskip#3\relax\H@svsec}{#8}%
968   }%
969   \@@par
970 \endgroup
971 \@ifundefined{#1smark}{\@gobble}{\csname #1smark\endcsname}{#7}%
972 \addcontentsline{toc}{#1}{\protect\numberline{#8}%
973 }{%
974 \def\@svsechd{%
975   #6{%
976     \@ifundefined{@runin@tos@#1}{\@runin@tos}{\csname @runin@tos@#1\endcsname}%
  Removed {\hskip#3\relax\H@svsec}{\Sectionformat{#8}{#1}}
977     {\hskip#3\relax\H@svsec}{#8}%
978   }%
979   \@ifundefined{#1smark}{\@gobble}{\csname #1smark\endcsname}{#7}%
980   \addcontentsline{toc}{#1}{\protect\numberline{#8}%
981 }%
982 }%
983 \@xsect{#5}%
984 }%
985 \def\@hang@froms#1#2{#1#2}%
986 \def\@runin@tos #1#2{#1#2}%

```

`\init@hyperref` Document classes that incorporate this package will be `hyperref-savvy`. (To accomplish this, we ensure that `\hyperanchor` and `\hyper@last` are both defined.) Being `hyperref-savvy` levels some requirements on us, but the benefits are many.

One is that the TOC will not get amnesia and require a full set of three typesetting runs before its formatting is stable. Instead, only two runs are required: the first updates the auxiliary file, the second the TOC. However, the formatting of the document does not change.

Another aspect of being `hyperref`-savvy is that the syntax of commands in the `.aux` file will not change if `hyperref` is turned on or off.

Note that `\hyper@anchorstart` and `\hyper@anchorend` constitute the programming interface for a hypertext anchor (the target of a hypertext link); `\hyper@linkstart` and `\hyper@linkend` are the interface for a hypertext link.

```

987 \def\init@hyperref{%
988 \providecommand\phantomsection{}%
989 \providecommand\hyper@makecurrent[1]{}%
990 \providecommand\Hy@raisedlink[1]{}%
991 \providecommand\hyper@anchorstart[1]{}%
992 \providecommand\hyper@anchorend{}%
993 \providecommand\hyper@linkstart[2]{}%
994 \providecommand\hyper@linkend{}%
995 \providecommand\@currentHref{}%
996 }%
997 \let\H@refstepcounter\refstepcounter
998 \appdef\document@inithook{%
999 \init@hyperref
1000 }%

```

`\sec@upcase` Upper case for sections (optional upper case items). These are created so that some headings can be toggled between mixed case and upper case readily. Headings that might be changed can be wrapped in the style file in `\sec@upcase{<text>}` constructs; the expansion of `\sec@upcase` is controlled here. It is `\relax` by default (mixed case heads), and can easily be changed to `\uppercase` if desired. If mixed-case headings are wanted by the editor, authors *must* supply mixed case text, although this is what authors should be doing anyway. (Mixed can be converted to upper, but the reverse transformation cannot be automated.)

The following setting gives the L^AT_EX default.

```

1001 \def\sec@upcase#1{\relax{#1}}%

```

6.14 Patch the tabular and array Environments

`\endtabular` We headpatch the begin processing and tailpatch the end processing of the `tabular` and `array` environments. A document class can define these hooks as needed.

We proceed with care to make further patches to support tabulars that break over pages. Our patches will not necessarily be effective for other packages that replace the L^AT_EX `array` and `tabular` environments. I know of none that do so.

```

1002 \appdef\document@inithook{%
1003 \ifpackageloaded{array}{\switch@array}{\switch@tabular}}%
1004 \prepdef\endtabular{\endtabular@hook}%
1005 \@provide\endtabular@hook{}%
1006 \prepdef\endarray{\endarray@hook}%
1007 \@provide\endarray@hook{}%
1008 \providecommand\array@hook{}%

```


Install, effectively, a head patch to `\tabular`. In order to avoid interference from, e.g., the `array` package, we must perform this patch only *after* packages load.

```
1009 \predef\@tabular{\tabular@hook}%
1010 \@provide\tabular@hook}%
1011 }%
```

`\switch@tabular` The two procedures `\switch@tabular` and `\switch@array` apply needed patches
`\switch@array` to the various tabular procedures, the former applying to the L^AT_EX kernel, the
latter to the required `array` package (and to the number of other required packages
that load it).

```
1012 \def\switch@tabular{%
1013   \let\@array@sw\@array@sw@array
1014   \@ifx{\@array\@array@LaTeX}{%
1015     \@ifx{\multicolumn\multicolumn@LaTeX}{%
1016       \@ifx{\@tabular\@tabular@LaTeX}{%
1017         \@ifx{\@tabarray\@tabarray@LaTeX}{%
1018           \@ifx{\array\array@LaTeX}{%
1019             \@ifx{\endarray\endarray@LaTeX}{%
1020               \@ifx{\endtabular\endtabular@LaTeX}{%
1021                 \@ifx{\@mkpream\@mkpream@LaTeX}{%
1022                   \@ifx{\@addamp\@addamp@LaTeX}{%
1023                     \@ifx{\@arrayacol\@arrayacol@LaTeX}{%
1024                       \@ifx{\@tabacol\@tabacol@LaTeX}{%
1025                         \@ifx{\@arrayclassz\@arrayclassz@LaTeX}{%
1026                           \@ifx{\@tabclassiv\@tabclassiv@LaTeX}{%
1027                             \@ifx{\@arrayclassiv\@arrayclassiv@LaTeX}{%
1028                               \@ifx{\@tabclassz\@tabclassz@LaTeX}{%
1029                                 \@ifx{\@classv\@classv@LaTeX}{%
1030                                   \@ifx{\hline\hline@LaTeX}{%
1031                                     \@ifx{\@tabularcr\@tabularcr@LaTeX}{%
1032                                       \@ifx{\@xtabularcr\@xtabularcr@LaTeX}{%
1033                                         \@ifx{\@xargarraycr\@xargarraycr@LaTeX}{%
1034                                           \@ifx{\@yargarraycr\@yargarraycr@LaTeX}{%
1035                                             \true@sw
1036                                           }{%
1037                                             \false@sw
1038                                           }%
1039                                         }{%
1040                                           \false@sw
1041                                         }%
1042                                       }{%
1043                                         \false@sw
1044                                       }%
1045                                     }{%
1046                                       \false@sw
1047                                     }%
1048                                   }{%
1049                                     \false@sw
1050                                   }%

```

```

1051         }{%
1052         \false@sw
1053         }%
1054     }{%
1055         \false@sw
1056         }%
1057     }{%
1058         \false@sw
1059         }%
1060     }{%
1061         \false@sw
1062         }%
1063     }{%
1064         \false@sw
1065         }%
1066     }{%
1067         \false@sw
1068         }%
1069     }{%
1070         \false@sw
1071         }%
1072     }{%
1073         \false@sw
1074         }%
1075     }{%
1076         \false@sw
1077         }%
1078     }{%
1079         \false@sw
1080         }%
1081     }{%
1082         \false@sw
1083         }%
1084     }{%
1085         \false@sw
1086         }%
1087     }{%
1088         \false@sw
1089         }%
1090     }{%
1091         \false@sw
1092         }%
1093     }{%
1094         \false@sw
1095         }%
1096     }{%
1097         \false@sw
1098         }%
1099     }{
1100 \class@info{Patching LaTeX tabular.}%

```

```

1101 }{%
1102 \class@info{Unrecognized LaTeX tabular. Please update this document class! (Proceeding with f
1103 }%
1104 \let\@array\@array@ltx
1105 \let\multicolumn\multicolumn@ltx
1106 \let\@tabular\@tabular@ltx
1107 \let\@tabarray\@tabarray@ltx
1108 \let\array\array@ltx
1109 \let\endarray\endarray@ltx
1110 \let\endtabular\endtabular@ltx
1111 \let\@mkpream\@mkpream@ltx
1112 \let\@addamp\@addamp@ltx
1113 \let\@arrayacol\@arrayacol@ltx
1114 \let\@tabacol\@tabacol@ltx
1115 \let\@arrayclassz\@arrayclassz@ltx
1116 \let\@tabclassiv\@tabclassiv@ltx
1117 \let\@arrayclassiv\@arrayclassiv@ltx
1118 \let\@tabclassz\@tabclassz@ltx
1119 \let\@classv\@classv@ltx
1120 \let\hline\hline@ltx
1121 \let\@tabularcr\@tabularcr@ltx
1122 \let\@xtabularcr\@xtabularcr@ltx
1123 \let\@xarraycr\@xarraycr@ltx
1124 \let\@yarraycr\@yarraycr@ltx
1125 }%

1126 \def\switch@array{%
1127 \@ifpackageloaded{colortbl}{\let\switch@array@info\colortbl@message}{\let\switch@array@info\ar
1128 \let\@array@sw\@array@sw@LaTeX
1129 \@ifx{\@array\@array@array}{%
1130 \@ifx{\@tabular\@tabular@array}{%
1131 \@ifx{\@tabarray\@tabarray@array}{%
1132 \@ifx{\array\array@array}{%
1133 \@ifx{\endarray\endarray@array}{%
1134 \@ifx{\endtabular\endtabular@array}{%
1135 \@ifx{\@mkpream\@mkpream@array}{%
1136 \@ifx{\@classx\@classx@array}{%
1137 \@ifx{\insert@column\insert@column@array}{%
1138 \@ifx{\@arraycr\@arraycr@array}{%
1139 \@ifx{\@xarraycr\@xarraycr@array}{%
1140 \@ifx{\@xarraycr\@xarraycr@array}{%
1141 \@ifx{\@yarraycr\@yarraycr@array}{%
1142 \true@sw
1143 }{%
1144 \false@sw
1145 }%
1146 }{%
1147 \false@sw
1148 }%
1149 }{%

```

```

1150         \false@sw
1151     }%
1152 }{%
1153     \false@sw
1154 }%
1155 }{%
1156     \false@sw
1157 }%
1158 }{%
1159     \false@sw
1160 }%
1161 }{%
1162     \false@sw
1163 }%
1164 }{%
1165     \false@sw
1166 }%
1167 }{%
1168     \false@sw
1169 }%
1170 }{%
1171     \false@sw
1172 }%
1173 }{%
1174     \false@sw
1175 }%
1176 }{%
1177     \false@sw
1178 }%
1179 }{%
1180     \false@sw
1181 }{%
1182 \class@info{Patching array package.}%
1183 }{%
1184 \switch@array@info
1185 }%
1186 \let\@array    \@array@array@new
1187 \let\@@array   \@array % Così fan tutti
1188 \let\@tabular  \@tabular@array@new
1189 \let\@tabarray \@tabarray@array@new
1190 \let\@array    \@array@array@new
1191 \let\endarray  \endarray@array@new
1192 \let\endtabular\endtabular@array@new
1193 \let\@mkpream  \@mkpream@array@new
1194 \let\@classx   \@classx@array@new
1195 \let\@arrayacol\@arrayacol@ltx
1196 \let\@tabacol  \@tabacol@ltx
1197 \let\insert@column\insert@column@array@new
1198 \expandafter\let\csname endtabular*\endcsname\endtabular % Così fan tutti
1199 \let\@arraycr  \@arraycr@new

```

```

1200 \let\@xarraycr \@xarraycr@new
1201 \let\@xargarraycr\@xargarraycr@new
1202 \let\@yargarraycr\@yargarraycr@new
1203 }%
1204 \def\array@message{%
1205 \class@info{Unrecognized array package. Please update this document class! (Proceeding with fi
1206 }%
1207 \def\colortbl@message{%
1208 \class@info{colortbl package is loaded. (Proceeding with fingers crossed.)}%
1209 }%

```

`\@array@sw` The Boolean `\@array@sw` must be different depending on whether the array package is loaded.

```

1210 \def\@array@sw@LaTeX{\@ifx{\@tabularcr}}%
1211 \def\@array@sw@array{\@ifx{\dollarbegin\beginingroup}}%

```

`\@tabular` We provide the old versions of `\@tabular` along with the respective new versions. The change here is to avoid committing to LR mode. That will be done later (as late as possible, naturally).

Compatibility note: I had done `\let \col@sep \@undefined` here, but this was not compatible with `colortbl`. I have removed that statement.

```

1212 \def\@tabular@LaTeX{%
1213 \leavevmode
1214 \hbox\bgroup$%
1215 \let\@acol\@tabacol
1216 \let\@classz\@tabclassz
1217 \let\@classiv\@tabclassiv
1218 \let\@tabularcr
1219 \@tabarray
1220 }%
1221 \def\@tabular@ltx{%
1222 \let\@acoll\@tabacoll
1223 \let\@acolr\@tabacolr
1224 \let\@acol\@tabacol
1225 \let\@classz\@tabclassz
1226 \let\@classiv\@tabclassiv
1227 \let\@tabularcr
1228 \@tabarray
1229 }%
1230 \def\@tabular@array{%
1231 \leavevmode
1232 \hbox\bgroup$%
1233 \col@sep\tabcolsep
1234 \let\dollarbegin\beginingroup
1235 \let\dollarend\endgroup
1236 \@tabarray
1237 }%
1238 \def\@tabular@array@new{%
1239 \let\@acoll\@tabacoll

```

```

1240 \let\@acolr\@tabacolr
1241 \let\@acol\@tabacol
      sepundefined
1242 \let\dollarbegin\begin
1243 \let\dollarend\end
1244 \@tabarray
1245 }%

```

`\@tabarray` Here we provide old and new versions of the `\@tabarray` procedure. The change here is to parametrize the default vertical alignment, which is 'c' in standard L^AT_EX. Under some circumstances, we want to change this to, say, 'v'.

FIXME: must decouple `array` and `tabular`. Done (it seems).

Note on `colortbl`: this package head-patches `\@tabarray` with its own command `\CTstart`, and tails onto `\endarray` with `\CTend`. It fortuitously does the former at `\AtBeginDocument` time, and, fortuitously, we do not patch `\endarray`, which it overwrites.

```

1246 \def\@tabarray@LaTeX{%
1247 \m@th\@ifnextchar[\@array{\@array[c]}%
1248 }%
1249 \def\@tabarray@ltx{%
1250 \m@th\@ifnextchar[\@array{\expandafter\@array\expandafter[\@array@default]}%
1251 }%
1252 \def\@tabarray@array{%
1253 \@ifnextchar[{\@@array}{\@@array[c]}%
1254 }%
1255 \def\@tabarray@array@new{%
1256 \@ifnextchar[{\@@array}{\expandafter\@array\expandafter[\@array@default]}%
1257 }%

```

`\@tabularcr` We provide for the `\@` command within `tabular` to provide control over page breaking, just the same as that of `eqnarray`.

`\@tabularcr` The count register `\intertabularlinepenalty` is similar to `\interdisplaylinepenalty`: it is the penalty associated with each row of a `tabular`. When it is set to `\@M`, the `tabular` will cleave together.

`\@xargarraycr` The count register `\@tbpen` is similar to `\@eqpen`: it memorizes the penalty to use after the current `tabular` row. If the `\@` command is in its star form, then `\@arraycr` `\@eqpen` is set to `\@M`.

We append code to `\samepage` so that a `tabular` within its scope will cleave together.

We keep the standard definition of `\@tabularcr` in `\@tabularcr@LaTeX` for reference, and provide a new definition that works like `\@eqnocr`: it sets `\@tbpen` to `\@M` if the star was given.

We also provide new versions of `\@xargarraycr`, `\@xargarraycr`, and `\@yargarraycr`, all of which invoke `\@tbpen`.

The `\switch@tabular` procedure switches in the new definitions.

```

1258 \newcount\intertabularlinepenalty
1259 \intertabularlinepenalty=100

```

```

1260 \newcount\@tbpen
1261 \appdef\samepage{\intertabularlinepenalty\M}%
1262 \def\@tabularcr@LaTeX{\ifnum 0='}\fi \ifstar \@xtabularcr \@xtabularcr}%
1263 \def\@tabularcr@ltx{\ifnum 0='}\fi \ifstar {\global \@tbpen \M \@xtabularcr }{\global \@tbpen
1264 \def\@xtabularcr@LaTeX{\ifnextchar [\@argtabularcr {\ifnum 0='{ \fi } \cr }}%
1265 \def\@xtabularcr@ltx{\ifnextchar [\@argtabularcr {\ifnum 0='{ \fi } \cr \noalign {\penalty \@tbpen
1266 \def\@xargarraycr@LaTeX#1{\@tempdima #1\advance \@tempdima \dp \@arstrutbox \vrule \@height \z@
1267 \def\@xargarraycr@ltx#1{\@tempdima #1\advance \@tempdima \dp \@arstrutbox \vrule \@height \z@
1268 \def\@yargarraycr@LaTeX#1{\cr \noalign {\vskip #1}}%
1269 \def\@yargarraycr@ltx#1{\cr \noalign {\penalty \@tbpen \vskip #1}}%

    If the array package has been loaded, we must alter the meanings of
    \@arraycr, \@xarraycr, \@xargarraycr, and \@yargarraycr. In this case, it
    is \switch@array that switches in the new definitions.

1270 \def\@arraycr@array{%
1271 \relax
1272 \iffalse{\fi\ifnum 0='}\fi
1273 \ifstar \@xarraycr \@xarraycr
1274 }%
1275 \def\@arraycr@new{%
1276 \relax
1277 \iffalse{\fi\ifnum 0='}\fi
1278 \ifstar {\global \@tbpen \M \@xarraycr }{\global \@tbpen \intertabularlinepenalty \@xarraycr
1279 }%
1280 \def\@xarraycr@array{%
1281 \@ifnextchar [%]
1282 \@argarraycr {\ifnum 0='{ \fi } \cr}%
1283 }%
1284 \def\@xarraycr@new{%
1285 \@ifnextchar [%]
1286 \@argarraycr {\ifnum 0='{ \fi } \cr \noalign {\penalty \@tbpen }}%
1287 }%
1288 \def\@xargarraycr@array#1{%
1289 \unskip
1290 \@tempdima #1\advance\@tempdima \dp\@arstrutbox
1291 \vrule \@depth\@tempdima \@width\z@
1292 \cr
1293 }%
1294 \def\@xargarraycr@new#1{%
1295 \unskip
1296 \@tempdima #1\advance\@tempdima \dp\@arstrutbox
1297 \vrule \@depth\@tempdima \@width\z@
1298 \cr
1299 \noalign {\penalty \@tbpen }}%
1300 }%
1301 \def\@yargarraycr@array#1{%
1302 \cr
1303 \noalign{\vskip #1}%
1304 }%
1305 \def\@yargarraycr@new#1{%

```

```

1306 \cr
1307 \noalign{\penalty \@tbpn \vskip #1}%
1308 }%

```

`\array` We provide old and new versions of the `\array` procedure for both L^AT_EX and the `array` package. The change here is to accomodate the new procedures that will be called for the array boundaries, even though at present they are not special. A thought: here is where matrices can be readily accomodated.

```

1309 \def\array@LaTeX{%
1310 \let\@acol\@arrayacol
1311 \let\@classz\@arrayclassz
1312 \let\@classiv\@arrayclassiv
1313 \let\@arraycr
1314 \let\@halignto\@empty
1315 \@tabarray
1316 }%
1317 \def\array@ltx{%
1318 \@ifmode{}{\@badmath$}%
1319 \let\@acoll\@arrayacol
1320 \let\@acolr\@arrayacol
1321 \let\@acol\@arrayacol
1322 \let\@classz\@arrayclassz
1323 \let\@classiv\@arrayclassiv
1324 \let\@arraycr
1325 \let\@halignto\@empty
1326 \@tabarray
1327 }%
1328 \def\array@array{%
1329 \col@sep\arraycolsep
1330 \def\d@llarbegin{$}\let\d@llarend\d@llarbegin\gdef\@halignto{}%
1331 \@tabarray
1332 }
1333 \def\array@array@new{%
1334 \@ifmode{}{\@badmath$}%
1335 \let\@acoll\@arrayacol
1336 \let\@acolr\@arrayacol
1337 \let\@acol\@arrayacol

```

Removed: `\let\col@sep\@undefined`

```

1338 \def\d@llarbegin{$}%
1339 \let\d@llarend\d@llarbegin
1340 \gdef\@halignto{}%
1341 \@tabarray
1342 }%

```

`\@array` Here we provide old and new versions of `\@array`. The change here is to provide a convenient, flexible, and extensible mechanism for new vertical alignment options. Instead of testing the optional argument with `\if`, we use a dispatcher based on `\csname`.

We also refrain from using `\ialign`, which would set the `\tabskip` to the wrong value.

Finally, the procedure to set the `\@arstrutbox` is broken out so that it can be patched.

```

1343 \def\@array@LaTeX[#1]#2{%
1344 \if #1\vtop \else \if#1b\vbox \else \vcenter \fi\fi
1345 \bgroup
1346 \setbox\@arstrutbox\hbox{%
1347 \vrule \@height\arraystretch\ht\strutbox
1348 \@depth\arraystretch \dp\strutbox
1349 \@width\z@}%
1350 \@mkpream{#2}%
1351 \edef\@preamble{%
1352 \ialign \noexpand\@halignto
1353 \bgroup \@arstrut \@preamble \tabskip\z@skip \cr}%
1354 \let\@startpbox\@startpbox \let\@endpbox\@endpbox
1355 \let\tabularnewline\%
1356 \let\par\@empty
1357 \let\@sharp##%
1358 \set@typeset@protect
1359 \lineskip\z@skip\baselineskip\z@skip
1360 \ifhmode \@preamerr\z@ \@@par\fi
1361 \@preamble
1362 }%
1363 \def\@array@ltx[#1]#2{%
1364 \@nameuse{\@array@align@#1}%
1365 \set@arstrutbox
1366 \@mkpream{#2}%
1367 \prepdef\@preamble{%
1368 \tabskip\tabmid@skip
1369 \@arstrut
1370 }%
1371 \appdef\@preamble{%
1372 \tabskip\tabright@skip
1373 \cr
1374 \array@row@pre
1375 }%
1376 % \let\@startpbox\@startpbox
1377 % \let\@endpbox\@endpbox
1378 \let\tabularnewline\%
1379 \let\par\@empty
1380 \let\@sharp##%
1381 \set@typeset@protect
1382 \lineskip\z@skip\baselineskip\z@skip
1383 \tabskip\tableft@skip\relax
1384 \ifhmode \@preamerr\z@ \@@par\fi
1385 \everycr{}%
1386 \expandafter\halign\expandafter\@halignto\expandafter\bgroup\@preamble
1387 }%

```

```

1388 %
1389 \def\set@arstrutbox{%
1390   \setbox\@arstrutbox\hbox{%
1391     \vrule \@height\arraystretch\ht\strutbox
1392           \@depth\arraystretch \dp\strutbox
1393           \@width\z@
1394   }%
1395 }%

```

\@array@array

```

1396 \def\@array@array[#1]#2{%
1397   \@tempdima \ht \strutbox
1398   \advance \@tempdima by\extrarowheight
1399   \setbox \@arstrutbox \hbox{\vrule
1400     \@height \arraystretch \@tempdima
1401     \@depth \arraystretch \dp \strutbox
1402     \@width \z@}%
1403   \begingroup
1404   \@mkpream{#2}%
1405   \xdef\@preamble{\noexpand \ialign \@halignto
1406     \bgroup \@arstrut \@preamble
1407     \tabskip \z@ \cr}%
1408   \endgroup
1409   \@arrayleft
1410   \if #1t\vtop \else \if#1b\vbox \else \vcenter \fi \fi
1411   \bgroup
1412   \let \@sharp ##\let \protect \relax
1413   \lineskip \z@
1414   \baselineskip \z@
1415   \m@th
1416   \let\\\@arraycr \let\tabularnewline\\\let\par\@empty \@preamble
1417 }%
1418 \def\@array@array@new[#1]#2{%
1419   \@tempdima\ht\strutbox
1420   \advance\@tempdima by\extrarowheight
1421   \setbox\@arstrutbox\hbox{%
1422     \vrule \@height\arraystretch\@tempdima
1423           \@depth \arraystretch\dp\strutbox
1424           \@width \z@
1425   }%
1426   \begingroup
1427   \@mkpream{#2}%
1428   \xdef\@preamble{\@preamble}%
1429   \endgroup
1430   \prepdef\@preamble{%
1431     \tabskip\tabmid@skip
1432     \@arstrut
1433   }%
1434   \appdef\@preamble{%
1435     \tabskip\tabright@skip

```

```

1436 \cr
1437 \array@row@pre
1438 }%
1439 \@arrayleft
1440 \@nameuse{@array@align@#1}%
1441 \m@th
1442 \let\\\@arraycr
1443 \let\tabularnewline\\%
1444 \let\par\@empty
1445 \let\@sharp##%
1446 \set@typeset@protect
1447 \lineskip\z@\baselineskip\z@
1448 \tabskip\tableft@skip
1449 \everycr{}%
1450 \expandafter\halign\expandafter\@halignto\expandafter\bgroup\@preamble
1451 }%

```

`\endarray` Here we provide old and new versions of `\endarray`. The change here is to use a single procedure to close out any array-like structure, namely `\endarray@ltx`. It merely closes out the `\halign`.

```

1452 \def\endarray@LaTeX{%
1453 \crr\egroup\egroup
1454 }%
1455 \def\endarray@ltx{%
1456 \crr\array@row@pst\egroup\egroup
1457 }%
1458 \def\endarray@array{%
1459 \crr \egroup \egroup \@arrayright \gdef\@preamble{}%
1460 }%
1461 \def\endarray@array@new{%
1462 \crr\array@row@pst\egroup\egroup % Same as \endarray@ltx
1463 \@arrayright
1464 \global\let\@preamble\@empty
1465 }%

```

`\endtabular`

```

1466 \def\endtabular@LaTeX{%
1467 \crr\egroup\egroup $\egroup
1468 }%
1469 \def\endtabular@ltx{%
1470 \endarray
1471 }%
1472 \def\endtabular@array{%
1473 \endarray $\egroup
1474 }%
1475 \def\endtabular@array@new{%
1476 \endarray
1477 }%

```

`endtabular*` Here we provide a proper definition for the star-form of `\end{endtabular}`. It is

one of the enduring curiosities that the L^AT_EX kernel continues to use dangerously and inappropriately “optimized” definitions for such commands.

```
1478 \@namedef{endtabular*}{\endtabular}%
```

`\multicolumn`

```
1479 \long\def\multicolumn@LaTeX#1#2#3{%
1480 \multispan{#1}\begingroup
1481 \@mkpream{#2}%
1482 \def\@sharp{#3}\set@typeset@protect
1483 \let\@startpbox\@startpbox\let\@endpbox\@endpbox
1484 \@arstrut \@preamble\hbox{}\endgroup\ignorespaces
1485 }%
1486 \long\def\multicolumn@ltx#1#2#3{%
1487 \multispan{#1}%
1488 \begingroup
1489 \@mkpream{#2}%
1490 \def\@sharp{#3}%
1491 \set@typeset@protect
1492 %\let\@startpbox\@startpbox\let\@endpbox\@endpbox
1493 \@arstrut
1494 \@preamble
1495 \hbox{}%
1496 \endgroup
1497 \ignorespaces
1498 }%
```

`\@array@align@` Here are the various procedures for the vertical alignment options. The change
`\array@default` from standard L^AT_EX is that we do not go into math mode in every case: only when required by `\vcenter`. Also, we use `\aftergroup` to close out the boxes and modes we have started. It requires only that each procedure issue exactly one unmatched `\bgroup`.

We establish here the default vertical alignment.

```
1499 \def\@array@align@t{\leavevmode\vtop\bgroup}%
1500 \def\@array@align@b{\leavevmode\vbox\bgroup}%
1501 \def\@array@align@c{\leavevmode\@ifmmode{\vcenter\bgroup}{\vcenter\bgroup\aftergroup$\aftergroup}
1502 \def\@array@align@v{%
1503 \@ifmmode{%
1504 \@badmath
1505 \vcenter\bgroup
1506 }{%
1507 \@ifinner{%
1508 $\vcenter\bgroup\aftergroup$
1509 }{%
1510 \@@par\bgroup
1511 }%
1512 }%
1513 }%
1514 \def\array@default{c}%
```

`\array@row@pre` The procedure `\array@row@rst` reestablishes a default context for an alignment, so that they can be nested. Any environment or procedure that alters the way alignments are formatted must patch this procedure to restore from that alteration. To start things off, we equate `\@array@align@v` to `\@array@align@c`, because it does not make sense to do the former in any context other than the MVL or in a list that will be unboxed onto the MVL.

```
1515 \def\array@row@rst{%
1516 \let\@array@align@v\@array@align@c
1517 }%
1518 \def\array@row@pre{%
1519 \def\array@row@pst{%
```

```
\toprule Default definitions for \toprule, \colrule, \botrule
\colrule 1520 \newcommand\toprule{\tab@rule{\column@font}{\column@fil}{\frstrut}}%
\botrule 1521 \newcommand\colrule{\unskip\lrstrut\\\tab@rule{\body@font}{\frstrut}}%
1522 \newcommand\botrule{\unskip\lrstrut\\\noalign{\hline@rule}{}}%
```

`\hline`

```
1523 \def\hline@LaTeX{%
1524 \noalign{\ifnum0='}\fi\hrule \@height \arrayrulewidth \futurelet
1525 \reserved@a\@xhline
1526 }%
1527 \def\hline@ltx{%
1528 \noalign{%
1529 \ifnum0='}\fi
1530 \hline@rule
1531 \futurelet\reserved@a\@xhline
1532 % \noalign ended in \@xhline
1533 }%
1534 \def\@xhline@unneeded{%
1535 \say\reserved@a
1536 \ifx\reserved@a\hline
1537 \vskip\doublerulesep
1538 \vskip-\arrayrulewidth
1539 \fi
1540 \ifnum0='}\fi}%
1541 }%
1542 \def\tab@rule#1#2#3{%
1543 \crr
1544 \noalign{%
1545 \hline@rule
1546 \gdef\@arstrut@hook{%
1547 \global\let\@arstrut@hook\@empty
1548 #3%
1549 }%
1550 \gdef\cell@font{#1}%
1551 \gdef\cell@fil{#2}%
1552 }%
1553 }%
```

```

1554 \def\column@font{}%
1555 \def\column@fil{}%
1556 \def\body@font{}%
1557 \def\cell@font{}%
1558 \def\frstrut{}%
1559 \def\lrstrut{}%

\@arstrut@hline The procedure \@arstrut@hline is substantially the same as \@arstrut, except
\@arstrut@org the strut copied in is \@arstrutbox@hline instead of \@arstrutbox.
\@arstrut@hook The procedure \@arstrut@hook is redefined in \tab@rule!
\@arstrutbox@hline The register \@arstrutbox@hline.
\set@arstrutbox We append to \set@arstrutbox the code necessary to set a strut following an
\hline@rule \hline.
                The procedure \hline@rule lays down a rule, and changes the meaning of
                \@arstrut so that the next line will be correctly strutted.
                The \@arstrut@hline@clnc is a klotch, a magic number.

1560 \def\@arstrut@hline{%
1561 \relax
1562 \@ifmmode{\copy}{\unhcopy}\@arstrutbox@hline
1563 \@arstrut@hook
1564 }%
1565 %
1566 \let\@arstrut@org\@arstrut
1567 \def\@arstrut@hook{%
1568 \global\let\@arstrut\@arstrut@org
1569 }%
1570 %
1571 \newbox\@arstrutbox@hline
1572 \appdef\set@arstrutbox{%
1573 \setbox\@arstrutbox@hline\hbox{%
1574 \setbox\z@\hbox{\$0^{0}_{}\$}%
1575 \dimen@ \ht\z@ \advance \dimen@ \@arstrut@hline@clnc
1576 \@ifdim{\dimen@ < \arraystretch \ht\strutbox}{\dimen@ = \arraystretch \ht\strutbox}{}%
1577 \vrule \@height \dimen@
1578 \depth \arraystretch \dp\strutbox
1579 \width \z@
1580 }%
1581 }%
1582 %
1583 \def\hline@rule{%
1584 \hrule \@height \arrayrulewidth
1585 \global\let\@arstrut\@arstrut@hline
1586 }%
1587 \def\@arstrut@hline@clnc{2\p}% % Klotch: magic number

\tableleft@skip
1588 \def\tableleft@skip{\z@skip}%
1589 \def\tablemid@skip{\z@skip}%\@flushglue
1590 \def\tableright@skip{\z@skip}%

```

```

1591 \def\tableftsep{\tabcolsep}%
1592 \def\tabmidsep{\tabcolsep}%
1593 \def\tabrightsep{\tabcolsep}%
1594 \def\cell@fil{}%
1595 \def\pbox@hook{}%

\@arstrut
1596 \appdef\@arstrut{\@arstrut@hook}%
1597 \let\@arstrut@hook\@empty
1598 \def\@addtopreamble{\appdef\@preamble}%

\@mkpream
1599 \def\@mkpream@LaTeX#1{%
1600   \@firstampttrue\@lastchclass6
1601   \let\@preamble\@empty
1602   \let\protect\@unexpandable@protect
1603   \let\@sharp\relax
1604   \let\@startpbox\relax\let\@endpbox\relax
1605   \@expast{#1}%
1606   \expandafter\@tfor \expandafter
1607     \@nextchar \expandafter:\expandafter=\reserved@a\do
1608     {\@testpach\@nextchar
1609     \ifcase \@chclass \@classz \or \@classi \or \@classii \or \@classiii
1610       \or \@classiv \or \@classv \fi\@lastchclass\@chclass}%
1611   \ifcase \@lastchclass \@acol
1612     \or \or \@preamerr \@ne\or \@preamerr \tw@\or \or \@acol \fi
1613 }%
1614 \def\@mkpream@ltx#1{%
1615   \@firstampttrue
1616   \@lastchclass6
1617   \let\@preamble\@empty
1618   \let\protect\@unexpandable@protect
1619   \let\@sharp\relax
1620   \let\@startpbox\relax\let\@endpbox\relax
1621   \@expast{#1}%
1622   \expandafter\@tfor \expandafter\@nextchar \expandafter:\expandafter=\reserved@a
1623   \do{%
1624     \expandafter\@testpach \expandafter{\@nextchar}%
1625     \ifcase \@chclass
1626       \@classz
1627       \or
1628       \@classi
1629       \or
1630       \@classii
1631       \or
1632       \@classiii
1633       \or
1634       \@classiv
1635       \or
1636       \@classv

```

```

1637 \fi
1638 \@lastchclass\@chclass
1639 }%
1640 \ifcase\@lastchclass
1641 \@acolr % right-hand column
1642 \or
1643 \or
1644 \@preamerr\@ne
1645 \or
1646 \@preamerr\tw@
1647 \or
1648 \or
1649 \@acolr % right-hand column
1650 \fi
1651 }%

```

`\insert@column`

```

1652 \def\insert@column@array{%
1653   \the@toks \the \@tempcnta
1654   \ignorespaces \@sharp \unskip
1655   \the@toks \the \count@ \relax
1656 }%
1657 \def\insert@column@array@new{%
1658   \the@toks\the\@tempcnta
1659   \array@row@rst\cell@font
1660   \ignorespaces\@sharp\unskip
1661   \the@toks\the\count@
1662   \relax
1663 }%

```

`\@mkpream@relax` The procedure `\@mkpream@relax` participates in a strange and wonderful method of binding the alignment procedure—but only certain parts thereof.

Here is how it works: in \LaTeX , the `array` package, and in the `longtable` package alike, there is a need to create an alignment preamble (using `\@mkpream`) for use by the upcoming `\halign`. Then, in both `array` and `longtable`, \TeX 's `\edef` is used to ‘compile in place’ that alignment preamble.

In the case of `array`, the operation is done in order to pre-expand the use of `*`; in `longtable`, it is to set the widths of the columns.

Now, during this `\edef`, certain control sequence names must *not* be expanded, and those are robustified by `\@mkpream@relax`.

```

1664 \def\@mkpream@relax{%
1665   \let\tableleftsep \relax
1666   \let\tabmidsep \relax
1667   \let\tabrightsep \relax
1668   \let\array@row@rst\relax
1669   \let\cell@font \relax
1670   \let\@startpbox \relax
1671 }%

```


`\@mkpream` We insert `\@mkpream@relax` at the head of the procedure. The robustifying of `\@startpbox` and `\@endpbox` is taken over by this mechanism. We also invoke `\@acolr` instead of `\@acol` when a right-hand column is at hand.

Note on `colortbl`: this package head-patches `\@mkpream` to robustify a number of its commands during the construction of the alignment preamble. The best we can do is to supplement the `\@mkpream@relax` procedure to perform this action.

```

1672 \def\@mkpream@array#1{%
1673   \gdef\@preamble{ }\@lastchclass 4 \@firstamptrue
1674   \let\@sharp\relax \let\@startpbox\relax \let\@endpbox\relax
1675   \@temptokena{#1}\@tempswattrue
1676   \@whilesw\if@tempswa\fi{\@tempswafalse\the\NC@list}%
1677   \count@m@ne
1678   \let\the@toks\relax
1679   \prepnext@tok
1680   \expandafter \@tfor \expandafter \@nextchar
1681   \expandafter :\expandafter =\the\@temptokena \do
1682   {\@testpach
1683   \ifcase \@chclass \@classz \or \@classi \or \@classii
1684     \or \save@decl \or \or \@classv \or \@classvi
1685     \or \@classvii \or \@classviii
1686     \or \@classx
1687     \or \@classx \fi
1688   \@lastchclass\@chclass}%
1689   \ifcase\@lastchclass
1690     \@acol \or
1691     \or
1692     \@acol \or
1693     \@preamerr \thr@@ \or
1694     \@preamerr \tw@ \@addtopreamble\@sharp \or
1695     \or
1696     \else \@preamerr \@ne \fi
1697   \def\the@toks{\the@toks}%
1698 }%
1699 \def\@mkpream@array@new#1{%
1700   \gdef\@preamble{ }%
1701   \@lastchclass\@f@ur
1702   \@firstamptrue
1703   \let\@sharp\relax
1704   \@mkpream@relax
1705   %\let\@startpbox\relax\let\@endpbox\relax
1706   \@temptokena{#1}\@tempswattrue
1707   \@whilesw\if@tempswa\fi{\@tempswafalse\the\NC@list}%
1708   \count@m@ne
1709   \let\the@toks\relax
1710   \prepnext@tok
1711   \expandafter \@tfor \expandafter \@nextchar \expandafter :\expandafter =\the\@temptokena
1712   \do{%
1713     \@testpach
1714     \ifcase\@chclass

```

```

1715 \@classz
1716 \or
1717 \@classi
1718 \or
1719 \@classii
1720 \or
1721 \save@decl
1722 \or
1723 \or
1724 \@classv
1725 \or
1726 \@classvi
1727 \or
1728 \@classvii
1729 \or
1730 \@classviii
1731 \or
1732 \@classx
1733 \or
1734 \@classx
1735 \fi
1736 \@lastchclass\@chclass
1737 }%
1738 \ifcase\@lastchclass
1739 \@acolr % right-hand column
1740 \or
1741 \or
1742 \@acolr % right-hand column
1743 \or
1744 \@preamerr\thr@@
1745 \or
1746 \@preamerr\tw@\@addtopreamble\@sharp
1747 \or
1748 \or
1749 \else
1750 \@preamerr\@ne
1751 \fi
1752 \def\the@toks{\the\toks}%
1753 }%

```

`\@mkpream@relax` David P. Carlisle's `colortbl` package headpatches `\@mkpream` in place during package loading, so it does not know whom it is working on. Let us try to accommodate this package by doing what it would have liked to have done.

Note: it would be far better to break out this mechanism in the `array` package.

```

1754 \appdef\@mkpream@relax{%
1755 \let\CT@setup \relax
1756 \let\CT@color \relax
1757 \let\CT@do@color \relax
1758 \let\color \relax

```

```

1759 \let\CT@column@color\relax
1760 \let\CT@row@color \relax
1761 \let\CT@cell@color \relax
1762 }%

\@addamp
1763 \def\@addamp@LaTeX{%
1764 \if@firstamp\@firstampfalse\else\edef\@preamble{\@preamble &}\fi
1765 }%
1766 \def\@addamp@ltx{%
1767 \if@firstamp\@firstampfalse\else\@addtopreamble{&}\fi
1768 }%

\@arrayacol
1769 \def\@arrayacol@LaTeX{%
1770 \edef\@preamble{\@preamble \hskip \arraycolsep}%
1771 }%
1772 \def\@arrayacol@ltx{%
1773 \@addtopreamble{\hskip\arraycolsep}%
1774 }%

\@tabacol
1775 \def\@tabacolI{%
1776 \@addtopreamble{\hskip\tableftsep\relax}%
1777 }%
1778 \def\@tabacol@LaTeX{%
1779 \edef\@preamble{\@preamble \hskip \tabcolsep}%
1780 }%
1781 \def\@tabacol@ltx{%
1782 \@addtopreamble{\hskip\tabmidsep\relax}%
1783 }%
1784 \def\@tabacolr{%
1785 \@addtopreamble{\hskip\tabrightsep\relax}%
1786 }%

\@arrayclassz
1787 \def\@arrayclassz@LaTeX{%
1788 \ifcase \@lastchclass \@acolampacol \or \@ampacol \or
1789 \or \or \@addamp \or
1790 \@acolampacol \or \@firstampfalse \@acol \fi
1791 \edef\@preamble{\@preamble
1792 \ifcase \@chnum
1793 \hfil$\relax\@sharp$\hfil \or $\relax\@sharp$\hfil
1794 \or \hfil$\relax\@sharp$\fi}%
1795 }%
1796 \def\@arrayclassz@ltx{%
1797 \ifcase\@lastchclass
1798 \@acolampacol
1799 \or

```

```

1800 \@ampacol
1801 \or
1802 \or
1803 \or
1804 \@addamp
1805 \or
1806 \@acolampacol
1807 \or
1808 \@firstampfalse\@acoll
1809 \fi
1810 \ifcase\@chnum
1811 \@addtopreamble{%
1812 \hfil\array@row@rst$\relax\@sharp$\hfil
1813 }%
1814 \or
1815 \@addtopreamble{%
1816 \array@row@rst$\relax\@sharp$\hfil
1817 }%
1818 \or
1819 \@addtopreamble{%
1820 \hfil\array@row@rst$\relax\@sharp$%
1821 }%
1822 \fi
1823 }%

```

\@tabclassz

```

1824 \def\@tabclassz@LaTeX{%
1825 \ifcase\@lastchclass
1826 \@acolampacol
1827 \or
1828 \@ampacol
1829 \or
1830 \or
1831 \or
1832 \@addamp
1833 \or
1834 \@acolampacol
1835 \or
1836 \@firstampfalse\@acol
1837 \fi
1838 \edef\@preamble{%
1839 \@preamble{%
1840 \ifcase\@chnum
1841 \hfil\ignorespaces\@sharp\unskip\hfil
1842 \or
1843 \hskip1sp\ignorespaces\@sharp\unskip\hfil
1844 \or
1845 \hfil\hskip1sp\ignorespaces\@sharp\unskip
1846 \fi}}%
1847 }%

```

```

1848 \def\@tabclassz@ltx{%
1849 \ifcase\@lastchclass
1850 \@acolampacol
1851 \or
1852 \@ampacol
1853 \or
1854 \or
1855 \or
1856 \@addamp
1857 \or
1858 \@acolampacol
1859 \or
1860 \@firstampfalse\@acoll
1861 \fi
1862 \ifcase\@chnum
1863 \@addtopreamble{%
1864   {\hfil\array@row@rst\cell@font\ignorespaces\@sharp\unskip\hfil}%
1865 }%
1866 \or
1867 \@addtopreamble{%
1868   {\cell@fil\hskip1sp\array@row@rst\cell@font\ignorespaces\@sharp\unskip\hfil}%
1869 }%
1870 \or
1871 \@addtopreamble{%
1872   {\hfil\hskip1sp\array@row@rst\cell@font\ignorespaces\@sharp\unskip\cell@fil}%
1873 }%
1874 \fi
1875 }%

```

\@tabclassiv

```

1876 \def\@tabclassiv@LaTeX{%
1877 \@addtopreamble\@nextchar
1878 }%
1879 \def\@tabclassiv@ltx{%
1880 \expandafter\@addtopreamble\expandafter{\@nextchar}%
1881 }%

```

\@arrayclassiv

```

1882 \def\@arrayclassiv@LaTeX{%
1883 \@addtopreamble{\@nextchar$}%
1884 }%
1885 \def\@arrayclassiv@ltx{%
1886 \expandafter\@addtopreamble\expandafter{\expandafter$\@nextchar$}%
1887 }%

```

\@classv

```

1888 \def\@classv@LaTeX{%
1889 \@addtopreamble{\@startpbox{\@nextchar}\ignorespaces
1890 \@sharp\@endpbox}%

```

```

1891 }%
1892 \def\@classv@ltx{%
1893 \expandafter\@addtopreamble
1894 \expandafter{%
1895 \expandafter \@startpbox
1896 \expandafter {\@nextchar}%
1897 \pbox@hook\array@row@rst\cell@font\ignorespaces\@sharp\@endpbox
1898 }%
1899 }%

```

`\@classx`

```

1900 \def\@classx@array{%
1901 \ifcase \@lastchclass
1902 \@acolampacol \or
1903 \@addamp \@acol \or
1904 \@acolampacol \or
1905 \or
1906 \@acol \@firstampfalse \or
1907 \@addamp
1908 \fi
1909 }%
1910 \def\@classx@array@new{%
1911 \ifcase \@lastchclass
1912 \@acolampacol
1913 \or
1914 \@addamp \@acol
1915 \or
1916 \@acolampacol
1917 \or
1918 \or
1919 \@firstampfalse\@acoll
1920 \or
1921 \@addamp
1922 \fi
1923 }%

```

6.15 Repair other broken parts of L^AT_EX

`\@xbitor` Expansion part has extraneous space token. Removed.

```

1924 \def\@xbitor@LaTeX #1{\@tempcntb \count#1
1925 \ifnum \@tempcnta =\z@
1926 \else
1927 \divide\@tempcntb\@tempcnta
1928 \ifodd\@tempcntb \@testtrue\fi
1929 \fi}%
1930 \def\@xbitor@ltx#1{%
1931 \@tempcntb\count#1\relax
1932 \@ifnum{\@tempcnta=\z@}{\fi}%
1933 \divide\@tempcntb\@tempcnta

```

```

1934 \@ifodd\@tempcntb{\@testtrue}{}%
1935 }%
1936 }%
1937 \@ifx{\@xbitor\@xbitor@LaTeX}{%
1938 \class@info{Repairing broken LaTeX \string\@xbitor}%
1939 }{%
1940 \class@info{Unrecognized LaTeX \string\@xbitor. Please update this document class! (Proceedin
1941 }%
1942 \let\@xbitor\@xbitor@ltx

```

6.16 Syntax

`\@gobble@opt@one` The `\@gobble@opt@one` command eats up an optional argument and one required argument.

```

1943 \newcommand*\@gobble@opt@one[2] [] {}%

```

6.17 Auto-indented Contents

Facility to automatically determine the proper indentation of the TOC entries.

Note on `hyperref` compatibility: We must respect that `\contentsline` now has a fourth argument. So, instead of trying to override the meaning of `\contentsline`, we use the aux file to remember max values from one run to the next.

In this respect, this package retains compatibility with `hyperref`.

`\@starttoc` Install hooks at beginning and end of the TOC processing.

```

1944 \def\@starttoc#1{%
1945 \begingroup
1946 \toc@pre
1947 \makeatletter
1948 \input{\jobname.#1}%
1949 \if@filesw
1950 \expandafter\newwrite\csname tf@#1\endcsname
1951 \immediate\openout \csname tf@#1\endcsname \jobname.#1\relax
1952 \fi
1953 \@nobreakfalse
1954 \toc@post
1955 \endgroup
1956 }%
1957 \def\toc@pre{%
1958 \def\toc@post{%

```

`\toc@@font` Interface for setting the formatting characteristics of this part of the TOC.

Note: `\toc@@font` is the common font for all auto-sizing toc commands, although this, too, could become a dispatcher.

```

1959 \def\toc@@font{}%
1960 \def\ltxu@dotsep{\z@}%

```

`\l@section` Interface for determining which TOC elements are automatically indented. All of the `\l@...` commands simply go through the utility procedure `\l@sections`. The calling convention is to pass the name of self and the name of parent. If you want to exclude any of these from the indentation scheme, simply leave the `\l@...` command undefined.

Note that the parent of “section” is nil, so we have to define a stub.

```

\def\l@section{\l@sections}{section}}% Implicit #3#4

\def\tocleft@{z}%

\def\l@subsection{\l@sections{section}{subsection}}% Implicit #3#4

\def\l@subsubsection{\l@sections{subsection}{subsubsection}}% Implicit #3#4

\def\l@paragraph{\l@sections{subsubsection}{paragraph}}% Implicit #3#4

\def\l@subparagraph#1#2{\l@sections{paragraph}{subparagraph}}% Implicit #3#4

```

Glom some `\dimen` registers.

```

1961 \let\tocdim@section      \leftmargini
1962 \let\tocdim@subsection  \leftmarginii
1963 \let\tocdim@subsubsection \leftmarginiii
1964 \let\tocdim@paragraph   \leftmarginiv
1965 \let\tocdim@appendix    \leftmarginv
1966 \let\tocdim@pagenum     \leftmarginvi

```

`\toc@pre@auto` We patch `\@starttoc` to: 1) before TOC processing, initialize the max registers and set the needed dimensions from the values stored in the auxiliary file, and 2) `\toc@post@auto` after TOC processing, store out those max register values into the auxiliary file.

Note that the font is set here: all other TOC entries must override these font settings.

To activate this override of the standard L^AT_EX processing, the substyle does:

`\let\toc@pre\toc@pre@auto` and `\let\toc@post\toc@post@auto`.

```

1967 \def\toc@pre@auto{%
1968   \toc@font
1969   \@tempdima\z@
1970   \toc@setindent\@tempdima{section}%
1971   \toc@setindent\@tempdima{subsection}%
1972   \toc@setindent\@tempdima{subsubsection}%
1973   \toc@setindent\@tempdima{paragraph}%
1974   \toc@letdimen{appendix}%
1975   \toc@letdimen{pagenum}%
1976 }%
1977 \def\toc@post@auto{%
1978   \if@filesw
1979   \begingroup
1980     \toc@writedimen{section}%
1981     \toc@writedimen{subsection}%
1982     \toc@writedimen{subsubsection}%

```



```

1983   \toc@writedimen{paragraph}%
1984   \toc@writedimen{appendix}%
1985   \toc@writedimen{pagenum}%
1986   \endgroup
1987   \fi
1988 }%

```

`\toc@setindent`

```

1989 \def\toc@setindent#1#2{%
1990   \csname tocdim@#2\endcsname\tocdim@min\relax
1991   \@ifundefined{tocmax@#2}{\@namedef{tocmax@#2}{\z@}}{}%
1992   \advance#1\@nameuse{tocmax@#2}\relax
1993   \expandafter\edef\csname tocleft@#2\endcsname{\the#1}%
1994 }%

```

`\toc@letdimen`

```

1995 \def\toc@letdimen#1{%
1996   \csname tocdim@#1\endcsname\tocdim@min\relax
1997   \@ifundefined{tocmax@#1}{\@namedef{tocmax@#1}{\z@}}{}%
1998   \expandafter\let\csname tocleft@#1\expandafter\endcsname\csname tocmax@#1\endcsname
1999 }%

```

`\toc@writedimen`

```

2000 \def\toc@writedimen#1{%
2001   \immediate\write\@auxout{%
2002     \gdef\expandafter\string\csname tocmax@#1\endcsname{%
2003       \expandafter\the\csname tocdim@#1\endcsname
2004     }%
2005   }%
2006 }%

```

`\l@sections` The procedure for formatting the indented TOC entries. We use control sequence names such as `\tocmax@section` and `\tocleft@section`, the former being written to the auxiliary file and the latter only defined for the duration of the TOC processing.

Note that the assignment of `\box\@tempboxa` by `\set@tocdim@pagenum` must endure over the invocation of #3: it contains the page number which will be set just before the `\par`.

The arguments:

- #1 superior section
- #2 this section
- #3 content, including possible `\numberline`
- #4 page number

```

2007 \def\l@sections#1#2#3#4{%
2008 \begingroup
2009 \everypar{}%
2010 \set@tocdim@pagenum\@tempboxa{#4}%
2011 \global\@tempdima\csname tocdim@#2\endcsname
2012 \leftskip\csname toclft@#2\endcsname\relax
2013 \dimen@\csname toclft@#1\endcsname\relax
2014 \parindent-\leftskip\advance\parindent\dimen@
2015 \rightskip\toclft@pagenum plus 1fil\relax
2016 \skip@\parfillskip\parfillskip\z@
2017 \let\numberline\numberline@@sections
2018 \nameuse{l@f@#2}%
2019 \ignorespaces#3\unskip\nobreak\hskip\skip@
2020 \hb@xt@\rightskip{\hfil\unhbox\@tempboxa}\hskip-\rightskip\hskip\z@skip

```

By side effect, set the value of, e.g., `\tocdim@section`.

Note that the `\par` must not be executed before the value of `\@tempdima` is expanded (outside the current group). Otherwise, the `lineno.sty` package may interfere (it unfortunately does a global assignment of `\@tempdima`).

```

2021 \expandafter\par
2022 \expandafter\aftergroup\csname tocdim@#2%
2023 \expandafter\endcsname
2024 \expandafter\endgroup
2025 \the\@tempdima\relax
2026 }%

```

In the call to `\set@tocdim@pagenum`, I am now exposing the use of the particular box register.

```

2027 \def\set@tocdim@pagenum#1#2{%
2028 \setbox#1\hbox{\ignorespaces#2}%
2029 \@ifdim{\tocdim@pagenum<\wd#1}{\global\tocdim@pagenum\wd#1}{-}%
2030 }%

```

`\numberline@@sections` The utility procedure for all `\numberline` processing in indented TOC entries. The first argument is self.

We use `\@tempdima` to pass a value around (via global assignment) because `\numberline` executes inside a group if the `hyperref` package is loaded. Would that it were not so!

```

2031 \def\numberline@@sections#1{%
2032 \leavevmode\hb@xt@-\parindent{%
2033 \hfil
2034 \@ifempty{#1}{-}{%
2035 \setbox\z@\hbox{#1.\kern\ltxu@dotsep}%
2036 \@ifdim{\@tempdima<\wd\z@}{\global\@tempdima\wd\z@}{-}%
2037 \unhbox\z@
2038 }%
2039 }%
2040 \ignorespaces
2041 }%
2042 \def\tocdim@min{\z@}%

```

6.18 Lists

`\list` Using `\parshape` to implement lists was always suspect (can you get behind `\parshape\@ne?`) and we now see that it was a mistake all along. Why? Because `\parshape`, like `\hangindent`, achieves its effect via “shifting” the `\hboxes` in a paragraph instead of using `\leftskip` and `\parindent`, which is robust during column balancing.

We introduce the alternative method with a hook into the L^AT_EX kernel procedure `\list`, which is the implementation of all lists.

```
2043 \def\list#1#2{%
2044   \ifnum \@listdepth >5\relax
2045     \@toodeep
2046   \else
2047     \global\advance\@listdepth\@ne
2048   \fi
2049   \rightmargin\z@
2050   \listparindent\z@
2051   \itemindent\z@
2052   \csname @list\romannumeral\the\@listdepth\endcsname
2053   \def\@itemlabel{#1}%
2054   \let\makelabel\@mklab
2055   \@nmbrolistfalse
2056   #2\relax
2057   \@trivlist
2058   \parskip\parsep
2059   \set@listindent
2060   \ignorespaces
2061 }%
2062 \def\set@listindent@parshape{%
2063   \parindent\listparindent
2064   \advance\@totalleftmargin\leftmargin
2065   \advance\linewidth-\rightmargin
2066   \advance\linewidth-\leftmargin
2067   \parshape\@ne\@totalleftmargin\linewidth
2068 }%
2069 \def\set@listindent@{%
2070   \parindent\listparindent
2071   \advance\@totalleftmargin\leftmargin
2072   \advance\rightskip\rightmargin
2073   \advance\leftskip\@totalleftmargin
2074 }%
2075 \let\set@listindent\set@listindent@parshape
```

6.19 Hypertext capabilities

`\href` We provide support for the `\href`, `\url`, and `\doi` commands. Packages, like `\url` `hyperref`, may override these definitions and provide better semantics.

```
\URL@prefix 2076 \providecommand\href[0]{\begingroup\@sanitize@url\@href}%
\doi 2077 \def\@href#1{\@startlink{#1}\endgroup\@@href}%
\doibase
```

```

2078 \def\@href#1{#1\@endlink}%
2079 \providecommand \url [0]{\begingroup\@sanitize@url \url }%
2080 \def \@url #1{\endgroup\@href {#1}{\URL@prefix#1}}%
2081 \providecommand \URL@prefix [0]{URL }%
2082 \providecommand\doi [0]{\begingroup\@sanitize@url\@doi}%
2083 \def\@doi#1{\endgroup\@startlink{\@doibase#1}doi:\discretionary {}{}#1\@endlink }%
2084 %changes{4.2a}{2017/11/21}{(MD) Use updated best practice to use https and doi.org}%
2085 \providecommand \doibase [0]{https://doi.org/}%
2086 \providecommand \@sanitize@url [0]{\chardef\cat@space\the\catcode'\@sanitize\catcode'\@cat@sp

```

`\@startlink` How we define `\@startlink` and `\@endlink` will depend on whether we are running under PDFLATEX. If so, and if PDF output is requested, then we use its primitives to implement hypertext, breaking out the link attributes in `\pdfstartlink@attr` and using the `hyperref` defaults; `\pdfstartlink@attr` can be redefined by a client package. Otherwise we fall back the HyperTeX standard and leave things to the DVI translator.

A class or package that wishes to employ hypertext capabilities should execute the `\hypertext@enable@ltx` procedure.

```

2087 \def\@startlink#1{%
2088 \def\@endlink{}%
2089 \@ifxundefined \pdfoutput {\true@sw}{\ifnum{\z@=\pdfoutput}{\true@sw}{\false@sw}}%
2090 {%
2091 \def\@startlink@hypertext#1{\leavevmode\special{html:<a href="#1">}}%
2092 \def\@endlink@hypertext{\special{html:</a>}}%
2093 }%
2094 \def\@startlink@hypertext#1{%
2095 \leavevmode
2096 \pdfstartlink\pdfstartlink@attr
2097 user{/Subtype/Link/A<</Type/Action/S/URI/URI(#1)>>}%
2098 \relax
2099 }%
2100 \def\@endlink@hypertext{\pdfendlink}%
2101 \def\pdfstartlink@attr{attr{/Border[0 0 1 ]/H/I/C[0 1 1]}}%
2102 }%
2103 \def\hypertext@enable@ltx{%
2104 \let\@startlink\@startlink@hypertext
2105 \let\@endlink\@endlink@hypertext
2106 }%

```

`\href` The `\href` command of `hyperref` was extend somewhere between versions 6.75r and 6.80e. We apply a repair to the earlier version (if present) so that it works like the later version.

The issue is the presence of whitespace, either following the `\href` token or following the first argument's closing brace character.

```

2107 \def\href@Hy{\hyper@normalise \href@ }%
2108 \def\href@Hy@ltx{\ifnextchar\bgroup\Hy@href{\hyper@normalise\href@}}%
2109 \def\Hy@href#\{\hyper@normalise\href@}%
2110 \begingroup

```

```

2111 \endlinechar=-1 %
2112 \catcode'\^^A=14 %
2113 \catcode'\^^M\active
2114 \catcode'\%\active
2115 \catcode'\#\active
2116 \catcode'\_\active
2117 \catcode'\$\active
2118 \catcode'\&\active
2119 \gdef\hyper@normalise@ltx{^^A
2120   \begingroup
2121   \catcode'\^^M\active
2122   \def^^M{ }^^A
2123   \catcode'\%\active
2124   \let%\@percentchar
2125   \let%\@percentchar
2126   \catcode'\#\active
2127   \def#{\hyper@hash}^^A
2128   \def\#{\hyper@hash}^^A
2129   \@makeother\^^A
2130   \edef&{\string&}^^A
2131   \edef&{\string&}^^A
2132   \edef\textunderscore{\string_}^^A
2133   \let\_ \textunderscore
2134   \catcode'\_\active
2135   \let\_ \textunderscore
2136   \let~\hyper@tilde
2137   \let~\hyper@tilde
2138   \let\textasciitilde\hyper@tilde
2139   \let\\\@backslashchar
2140   \edef${\string$}^^A
2141   \Hy@safe@activestrue
2142   \hyper@normalise
2143 }^^A
2144 \catcode'\#=6 ^^A
2145 \gdef\Hy@ActiveCarriageReturn@ltx{^^M}^^A
2146 \gdef\hyper@normalise@ltx#1#2{^^A
2147   \def\Hy@tempa{#2}^^A
2148   \ifx\Hy@tempa\Hy@ActiveCarriageReturn
2149     \Hy@ReturnAfterElseFi{^^A
2150       \hyper@@normalise{#1}^^A
2151     }^^A
2152   \else
2153     \Hy@ReturnAfterFi{^^A
2154       \hyper@@normalise{#1}{#2}^^A
2155     }^^A
2156   \fi
2157 }^^A
2158 \gdef\hyper@@normalise@ltx#1#2{^^A
2159   \def\Hy@tempa{^^A
2160     \endgroup

```

```

2161     \noexpand#1{\Hy@RemovePercentCr#2%^M@nil}^^A
2162   }^^A
2163   \Hy@tempa
2164 }^^A
2165 \gdef\Hy@RemovePercentCr@ltx#1%^M#2\@nil{^^A
2166   #1^^A
2167   \ifx\limits#2\limits
2168   \else
2169     \Hy@ReturnAfterFi{^^A
2170     \Hy@RemovePercentCr #2\@nil
2171   }^^A
2172   \fi
2173 }^^A
2174 \endgroup
2175 \def\switch@hyperref@href{%
2176 \expandafter\ifx\expandafter{\csname href \endcsname\href@Hy}{
2177 \class@info{Repairing hyperref 6.75r \string\href}%
2178 \let\hyper@normalise\hyper@normalise@ltx
2179 \let\hyper@@normalise\hyper@@normalise@ltx
2180 \let\hyper@n@rmalise\hyper@n@rmalise@ltx
2181 \let\Hy@ActiveCarriageReturn\Hy@ActiveCarriageReturn@ltx
2182 \let\Hy@RemovePercentCr\Hy@RemovePercentCr@ltx
2183 \let\href\href@Hy@ltx
2184 }{}}%
2185 }%
2186 \appdef\document@inithook{\switch@hyperref@href}%

```

\typeout We make the \typeout procedure of L^AT_EX be \long, because sometimes we are talking about \par.

```

2187 \def\typeout@org#1{%
2188 \begingroup
2189 \set@display@protect
2190 \immediate\write\@unused{#1}%
2191 \endgroup
2192 }%
2193 \long\def\typeout@ltx#1{%
2194 \begingroup
2195 \set@display@protect
2196 \immediate\write\@unused{#1}%
2197 \endgroup
2198 }%
2199 \@ifx{\typeout\typeout@org}{%
2200 \class@info{Making \string\typeout\space \string\long}%
2201 \let\typeout\typeout@ltx
2202 }{}}%

```

6.20 End of the kernel DOCSTRIP module

Here ends the module.

2203 %</kernel>