A LaTeX Package of utility macros *

Arthur Ogawa (mailto:ogawa@teleport.com), 1.0rc5b

August 3, 2001

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

The distribution point for this work is ftp://ftp.teleport.com/users/ogawa/macros/latex/contrib/supported/ltxutil..., which contains fully unpacked, prebuilt runtime files and documentation.

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

To use this document class, you must have a working TeX installation equipped with LaTeX 2ε 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 LaTeX.

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

Contents

*This file has version number 1.0rc5b, last revised 2001/07/31. For version number and date, search on ”1.0rc5b” in the .dtx file, or see the end of the 00readme.txt file.

1 Processing Instructions

The package file `ltxutil.sty` is generated from this file, `ltxutil.dtx`, using the DOCSTRIP facility of LaTeXvia `tex ltxutil.ins`. 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 2ε (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 installer `ltxutil.ins`, and the file `00readme.txt`.

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

2. Now typeset `ltxutil.ins`, thereby generating the package file `ltxutil.sty`.
3. Install `ltxutil.sty` by moving it to a location in your filesystem where they will be found by LaTeX.

1.2 Bill of Materials

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

1.2.1 Primary Source

One single file generates all.

```
%ltxutil.dtx
%
```

1.2.2 Generated by `latex ltxutil.dtx`

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

```
%00readme.txt    ltxutil.ins
%
```

1.2.3 Generated by `tex ltxutil.ins`

Typesetting the installer generates the package files.

```
%ltxutil.sty
%
```

1.2.4 Documentation

The following are the online documentation:

```
%ltxutil.pdf
%
```

1.2.5 Auxiliary

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

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

2 Code common to all modules

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

Now we declare that these files can only be used with LaTeX 2ε. An appropriate message is displayed if a different TeX format is used.

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

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

```
%<ltxutil>\ProvidesFile{ltxutil.sty}%
%<*doc>
\expandafter\ProvidesFile\expandafter{\jobname.dtx}%
%</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.

```
%<*doc|ltxutil>
 [2001/07/31 1.0rc5b utilities package]% \fileversion
%</doc|ltxutil>
```

3 The driver module doc

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

```
%<*doc>
```

3.1 The Preamble

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

```
\documentclass{ltxdoc}
\RequirePackage{ltxdocext}%
\let\url\undefined
\RequirePackage[colorlinks=true,linkcolor=blue]{hyperref}%
\expandafter\ifx\csname package@font\endcsname\@undefined\else
 \expandafter\RequirePackage\expandafter{\csname package@font\endcsname}%
\fi
\CodelineIndex\EnableCrossrefs
```

3.1.1 Docstrip and info directives

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

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

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

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

3.2 The installer file

The installer `ltxutil.ins` appears here. If you have retrieved the standard distribution of this package, the installer file is already on your filesystem. If you are bootstrapping, the first typesetting of the `.dtx` file will cause the installer to be generated.

The following modules are used to direct DOCSTRIP in generating the external files:

Module	File	Description
doc	`ltxutil.drv`	driver for programmer's documentation
ltxutil,ltxutil-krn	`ltxutil.sty`	this package
ltxutil-krn		the portion of this package suitable for inclusion within another package

```
\begin{filecontents}{ltxutil.ins}
%% This file will generate documentation and runtime files
%% from ltxutil.dtx when run through LaTeX or TeX.
\input docstrip
\preamble

This is a generated file;
altering it directly is inadvisable;
instead, modify the original source file.
See the URL in the file 00readme.txt.

Copyright notice.

   These files are distributed
   WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

\endpreamble
\keepsilent
  \generate{%
   \file{ltxutil.drv}{\from{ltxutil.dtx}{doc}}%
   \file{ltxutil.sty}{%
    \from{ltxutil.dtx}{ltxutil,ltxutil-krn}%
   }%
  }%
\ifToplevel{
\Msg{************************************************************}
\Msg{*}
\Msg{* To finish the installation, please move}
\Msg{*    ltxutil.sty}
\Msg{* into a directory searched by TeX.}
\Msg{*}
```

```
\Msg{* To produce the documentation,
       run ltxutil.dtx through LaTeX.}
\Msg{*}
\Msg{* Happy TeXing}
\Msg{*************************************************************}
}
\endbatchfile
\end{filecontents}
```

Note that, because all of the files generated by the installer are part of the standard distribution, it will be necessary to run the installer only when bootstrapping (or, of course, during development). Note, too, that it is rare to generate the `doc` module because it suffices to simply typeset the `.dtx` file itself.

3.3 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 beginining of the programmer's documentation. What better thing, after all, to have appear at the beginning of the typeset documentation?

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

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

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

```
\begin{filecontents*}{00readme.txt}
\title{%
 A \LaTeX\ Package of utility macros%
 \thanks{%
  This file has version number \fileversion,
  last revised \filedate.%
  % For version number and date,
  % search on "\fileversion" in the .dtx file,
  % or see the end of the 00readme.txt file.
 }%
}%

\author{%
Arthur Ogawa (\texttt{mailto:ogawa@teleport.com}),
\fileversion\\Copyright (C) 1999 Arthur Ogawa
}%
\maketitle

This file embodies the \classname{ltxutil} package,
the implementation and its user documentation.

The distribution point for this work is
\url{ftp://ftp.teleport.com/users/ogawa/macros/latex/contrib/supported/ltxutil...}
which contains fully unpacked, prebuilt runtime files and documentation.

The \classname{ltxutil} package was commissioned by the American Physical Society
and is distributed under the terms of the \LaTeX\ Project Public License,
the same license under which all the portions of \LaTeX\ itself is distributed.
```

```
Please see \url{http://ctan.tug.org/macros/latex/base/lppl.txt} for details.

To use this document class, you must have a working
\TeX\ installation equipped with \LaTeXe\
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 \file{ltxutil.sty}
into a location in your filesystem where it will be found by \LaTeX.

To use, read the user documentation \file{ltxutil.pdf}.

\tableofcontents

\section{Processing Instructions}

The package file \file{ltxutil.sty}
is generated from this file, \file{ltxutil.dtx},
using the {\sc docstrip} facility of \LaTeX
via |tex ltxutil.ins|.
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|.

\subsection{Build Instructions}

You may bootstrap this suite of files solely from \file{ltxutil.dtx}.
Prepare by installing \LaTeXe\ (and either tex or pdftex) on your computer,
then carry out the following steps:
\begin{enumerate}
\item
Within an otherwise empty directory,
typeset \file{ltxutil.dtx} with \LaTeX\ or pdflatex;
you will obtain the typeset documentation you are now reading,
along with
the installer \file{ltxutil.ins},
and the file \file{00readme.txt}.

Note: you will have to run \LaTeX\ twice, then \file{makeindex}, then
\LaTeX\ again in order to obtain a valid index and table of contents.
\item
Now typeset \file{ltxutil.ins},
thereby generating the package file \file{ltxutil.sty}.
\item
Install \classname{ltxutil.sty}
by moving it to a location
in your filesystem where they will be found by \LaTeX.
\end{enumerate}
\end{filecontents*}
```

3.4 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.

```
\begin{document}%
\expandafter\DocInput\expandafter{\jobname.dtx}%
% ^^A\PrintChanges
\end{document}
%</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 acccess to certain procedures, usually to be invoked by a LaTeX command or environment, but not at the document level.

5 Compatability 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 priviledged 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 compatability, `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 `ltxutil` DOCSTRIP module

```
%<*ltxutil>
\def\package@name{ltxutil}%
\expandafter\PackageInfo\expandafter{\package@name}{%
 Utility macros for \protect\LaTeXe,
 by A. Ogawa (ogawa@teleport.com)%
}%
%</ltxutil>
```

6.2 Banner

Credit where due.

```
%<*ltxutil-krn>
\typeout{%
 ltxutil: portions licensed from W. E. Baxter (web@superscript.com)%
}%
```

6.3 Errors and warnings

\class@err
\class@warn
\class@info

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

```
\def\class@err#1{\ClassError{\class@name}{#1}\@eha}%
\def\class@warn#1{\ClassWarningNoLine{\class@name}{#1}}%
\def\class@info#1{\ClassInfo{\class@name}{#1}}%
\def\obsolete@command#1{%
 \class@warn@end{Command \string#1\space is obsolete.^^JPlease remove from your do
 \global\let#1\@empty
 #1%
}%
\def\replace@command#1#2{%
 \class@warn@end{Command \string#1\space is obsolete;^^JUse \string#2\space instea
 \global\let#1#2%
```

```
 #1%
}%
\def\replace@environment#1#2{%
 \class@warn@end{Environment #1 is obsolete;^^JUse #2 instead}%
 \glet@environment{#1}{#2}%
 \@nameuse{#1}%
}%
\def\incompatible@package#1{%
 \@ifpackageloaded{#1}{%
  \def\@tempa{I cannot continue. You must remove the \string\usepackage\ statement
  \ClassError{\class@name}{The #1 package cannot be used with \class@name}%
  \@tempa\stop
 }{%
  \class@info{#1 was not loaded (OK!)}%
 }%
}%
\def\class@warn@end#1{%
 \gappdef\class@enddocumenthook{\class@warn{#1}}%
}%
\AtEndOfClass{%
 \@ifxundefined\class@name{\def\class@name{Generic Class}}{}%
}%
```

6.4 New Tools

`\t@`

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

`\dimen@iii`

```
\dimendef\dimen@iii\thr@@
```

`\halignt@`

```
\def\halignt@{\halign\t@}%
```

`\f@ur` Analogous to `\@ne`, `\tw@`, and `\thr@@`.

```
\chardef\f@ur=4\relax
\chardef\cat@letter=11\relax
\chardef\other=12\relax
```

`\let@environment` `\glet@environment` The directive `\let@environment` takes care of a common programming idiom whereby one environment is made a synonym for another.

```
\def\let@environment#1#2{%
 \expandafter\let
 \csname#1\expandafter\endcsname\csname#2\endcsname
 \expandafter\let
 \csname end#1\expandafter\endcsname\csname end#2\endcsname
}%
\def\glet@environment#1#2{%
 \global\expandafter\let
 \csname#1\expandafter\endcsname\csname#2\endcsname
 \global\expandafter\let
 \csname end#1\expandafter\endcsname\csname end#2\endcsname
}%
```

\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.

```
\newcommand\tracingplain{%
 \tracingonline\z@\tracingcommands\z@\tracingstats\z@
 \tracingpages\z@\tracingoutput\z@\tracinglostchars\@ne
 \tracingmacros\z@\tracingparagraphs\z@\tracingrestores\z@
 \showboxbreadth5\showboxdepth3\relax %\errorstopmode
 }%
\newcommand\traceoutput{%
 \appdef\@resetactivechars{\showoutput}%
}%
```

\say \saythe The commands \say and \saythe cause diagnostic messages in the TEX log that give the value of a control sequence name or a register respectively.

```
\newcommand\say[1]{\typeout{<\noexpand#1=\meaning#1>}}%
\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 \baselinesklip mechanism. Can be invoked just after a \hrule to undo its default suppression of base line skip.

```
\def\fullinterlineskip{\prevdepth\z@}%
```

\count@i \count@ii

```
\countdef\count@i\@ne
\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@superscript.com). See articles by Baxter and Ogawa in the proceedings of the 1994 TUG meeting, TUGboat Vol. 15, No. 3.

\prepdef \appdef \gappdef Provide the capability of performing head- and tail patches. The procedure \prepdef prepends to the given macro the tokens specified in its second argument. 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.

```
\long\def\prepdef#1#2{%
 \@ifxundefined#1{\toks@{}}{\toks@\expandafter{#1}}%
 \toks@ii{#2}%
 \edef#1{\the\toks@ii\the\toks@}%
}%
\long\def\appdef#1#2{%
 \@ifxundefined#1{\toks@{}}{\toks@\expandafter{#1}}%
 \toks@ii{#2}%
 \edef#1{\the\toks@\the\toks@ii}%
}%
\long\def\gappdef#1#2{%
 \@ifxundefined#1{\toks@{}}{\toks@\expandafter{#1}}%
 \toks@ii{#2}%
```

```
 \global\edef#1{\the\toks@\the\toks@ii}%
}%
\long\def\appdef@val#1#2{%
 \appdef#1{{#2}}%
}%
\long\def\appdef@e#1#2{%
 \expandafter\appdef
 \expandafter#1%
 \expandafter{#2}%
}%
\long\def\appdef@eval#1#2{%
 \expandafter\appdef@val
 \expandafter#1%
 \expandafter{#2}%
}%
\toksdef\toks@ii=\tw@
```

\@ifxundefined
\@ifnotrelax
\@argswap
\@argswap@val

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

The procedures \@argswapand \@argswap@valare used to facilitate control of expansion.

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

\@boolean
\@boole@def

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

```
\def\@boolean#1#2{%
  \long\def#1{%
    #2% \if<something>
      \expandafter\true@sw
    \else
      \expandafter\false@sw
    \fi
  }%
}%
\def\@boole@def#1#{\@boolean{#1}}% Implicit #2
```

\@booleantrue
\@booleanfalse

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

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

\@ifx
\@ifx@empty
\@if@empty
\@ifcat
\@ifdim
\@ifeof
\@ifhbox
\@ifhmode
\@ifinner
\@ifmmode
\@ifnum
\@ifodd
\@ifvbox
\@ifvmode
\@ifvoid

We can now invoke the defining word to create the procedures \@ifx and friends.

Compatability Note: earlier versions of this package defined a procedure \@ifempty. However, for compatability with AMSLaTeX, we must avoid the following three names: \@ifempty, \@xifempty, and \@ifnotempty.

```
\@boole@def\@ifx#1{\ifx#1}%
\@boole@def\@ifx@empty#1{\ifx\@empty#1}%
```

```
\@boole@def\@if@empty#1{\if!#1!}%
%\@boole@def\@if@sw#1{\csname if#1\endcsname}%
\def\@if@sw#1#2{#1\expandafter\true@sw\else\expandafter\false@sw#2}%
\@boole@def\@ifdim#1{\ifdim#1}%
\@boole@def\@ifeof#1{\ifeof#1}%
\@boole@def\@ifhbox#1{\ifhbox#1}%
\@boole@def\@ifhmode{\ifhmode}%
\@boole@def\@ifinner{\ifinner}%
\@boole@def\@ifmmode{\ifmmode}%
\@boole@def\@ifnum#1{\ifnum#1}%
\@boole@def\@ifodd#1{\ifodd#1}%
\@boole@def\@ifvbox#1{\ifvbox#1}%
\@boole@def\@ifvmode{\ifvmode}%
\@boole@def\@ifvoid#1{\ifvoid#1}%
```

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

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

`\loopuntil` `\loopwhile` 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.

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

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

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

6.6 Begin Document Structure

The standard LaTeX 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, LaTeX 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`.

The price we pay for this facility is to depend on the stability of this part of LaTeX's kernel code (the first token of `\document`), which could change, you see. But considering that LaTeX 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`.

```
\prepdef\document{%
 \endgroup
```

```
 \init@documenthook
 \set@typesize@hook
 \normalsize
 \set@pica@hook
 \true@sw{}%
}%
```

`\class@documenthook`
`\class@enddocumenthook`

We install the first `\AtBeginDocument` hook, namely the procedure `\class@documenthook`. 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` and `\appdef\class@enddocumenthook` instead of `\AtBeginDocument` and `\AtEndDocument`.

```
\def\init@documenthook{}%
\AtBeginDocument{%
 \class@documenthook
}%
\AtEndDocument{%
 \class@enddocumenthook
}%
\def\class@documenthook{}%
\def\class@enddocumenthook{}%
```

`\set@typesize@hook`
`\set@pica@hook`

The macros `\set@typesize@hook` and `\set@pica@hook` provide everything we need. To use, simply `\appdef` your tokens to the appropriate hook.

```
\def\set@typesize@hook{}%
\def\set@pica@hook{}%
```

`\enddocument`
`\check@aux`
`\do@check@aux`

The standard LaTeX `\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.

```
\def\enddocument{%
 \@enddocumenthook
 \@checkend{document}%
 \clear@document
 \check@aux
 \deadcycles\z@
 \@@end
}%
\def\clear@document{\clearpage}%
\def\check@aux{\do@check@aux}%
\def\do@check@aux{%
 \@if@sw\if@filesw\fi{%
  \immediate\closeout\@mainaux
  \let\@setckpt\@gobbletwo
  \let\@newl@bel\@testdef
  \@tempswafalse
  \makeatletter
  \input\jobname.aux\relax
```

```
}{}%
\@dofilelist
\@ifdim{\font@submax >\fontsubfuzz\relax}{%
 \@font@warning{%
  Size substitutions with differences\MessageBreak
  up to \font@submax\space have occured.\@gobbletwo
 }%
}{}%
\@defaultsubs
\@refundefined
\@if@sw\if@filesw\fi{%
 \@ifx{\@multiplelabels\relax}{%
  \@if@sw\if@tempswa\fi{%
   \@latex@warning@no@line{%
    Label(s) may have changed.
    Rerun to get cross-references right
   }%
  }{}%
 }{%
   \@multiplelabels
 }%
}{}%
}%
```

6.7 Type Tools

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

```
\def\flushing{%
  \let\\\@normalcr
  \leftskip\z@skip
  \rightskip\z@skip
  \@rightskip\z@skip
  \parfillskip\@flushglue
}%
```

6.8 Display Math

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

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

```
\def\eqnarray@LaTeX{%
   \stepcounter{equation}%
   \def\@currentlabel{\p@equation\theequation}%
   \global\@eqnswtrue
   \m@th
   \global\@eqcnt\z@
   \tabskip\@centering
   \let\\\@eqncr
   $$\everycr{}\halign\t@\displaywidth\bgroup
       \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnsel
```

```
      &\global\@eqcnt\@ne\hskip \tw@\arraycolsep \hfil${##}$\hfil
      &\global\@eqcnt\tw@ \hskip \tw@\arraycolsep
         $\displaystyle{##}$\hfil\tabskip\@centering
      &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
         \tabskip\z@skip
      \cr
}
\long\def\eqnarray@fleqn@fixed{%
 \stepcounter{equation}\def\@currentlabel{\p@equation\theequation}%
 \global\@eqnswtrue\m@th\global\@eqcnt\z@
 \tabskip\mathindent
 \let\\=\@eqncr
 \setlength\abovedisplayskip{\topsep}%
 \ifvmode\addtolength\abovedisplayskip{\partopsep}\fi
 \addtolength\abovedisplayskip{\parskip}%
 \setlength\belowdisplayskip{\abovedisplayskip}%
 \setlength\belowdisplayshortskip{\abovedisplayskip}%
 \setlength\abovedisplayshortskip{\abovedisplayskip}%
 $$%
 \everycr{}%
 \halignt@\linewidth\bgroup
  \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnsel
  &\global\@eqcnt\@ne
   \hskip\tw@\eqncolsep
   \hfil${{}##{}}$\hfil
  &\global\@eqcnt\tw@
   \hskip\tw@\eqncolsep
   $\displaystyle{##}$\hfil\tabskip\@centering
  &\global\@eqcnt\thr@@\hb@xt@\z@\bgroup\hss##\egroup
   \tabskip\z@skip
  \cr
}%
\@ifx{\eqnarray\eqnarray@LaTeX}{%
 \class@info{Repairing broken LaTeX eqnarray}%
 \let\eqnarray\eqnarray@fleqn@fixed
 \newlength\eqncolsep
 \setlength\eqncolsep\z@
 \let\eqnarray@LaTeX\relax
 \let\eqnarray@fleqn@fixed\relax
}{}%
\def\mathindent{\@centering}%
\def\set@eqnarray@skips{}%
```

6.9 Footnotes

\footnote
\footnotemark
\@xfootnote
\@xfootnotemark
\@yfootnote

We repair an error in the LaTeX kernel (see `ltfloat.dtx`) involving footnotes. The symptom is that the `\footnotemark` command does not work properly within a `minipage` environment. The source of the problem is in the way the `\footnotemark` and `\@xfootnotemark` procedures are defined: they do not share the method used by the `\footnote` and other procedures that allows a context switch to change the way footnotes behave within a minipage environment. This is a LaTeX bug of long standing; this fix dates to 1987.

While we are at it, we rewrite both the `\footnote` and `\footnotemark` pro-

cedures, achieving a slightly cleaner separation of syntax and semantics. Note that the `\@footnotemark` and `\@footnotetext` procedures are not altered here; they continue as the methods of formatting the footnote mark and footnote text, respectively.

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\@footnotetext\@mpfootnotetext
%\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.

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

```
\def\footnote{%
 \@ifnextchar[\@xfootnote{\@yfootnote\@footnotetext}%
}%
\def\footnotemark{%
 \@ifnextchar[\@xfootnotemark{\@yfootnote}%
}%
\def\@xfootnote[#1]{%
 \@xfootnotemark[#1]%
 \@footnotetext
}%
\def\@xfootnotemark@ltx[#1]{%
 \begingroup
   \csname c@\@mpfn\endcsname #1\relax
   \unrestored@protected@xdef\@thefnmark{\thempfn}%
 \endgroup
 \H@@footnotemark
}%
\def\@yfootnote{%
 \stepcounter\@mpfn
 \protected@xdef\@thefnmark{\thempfn}%
 \H@@footnotemark
}%
```

Note on `hyperref` compatability: In its "Automated LaTeX hypertext cross-references", the `hyperref` package alters footnote processing, thereby imperiling these fixes and necessiating defensive measures.

The main thing `hyperref` does is to take over the `\@mpfootnotetext` and `\@footnotetext` procedures, feeding its own arguments to these macros. It also rewrites `\@footnotemark`, making it a hyperlink.

But at the same time, it attempts to turn off these changes during `\maketitle` processing, necessitating rewriting `\@xfootnotemark`. At this point it is on the slippery slope.

We make ourself `hyperref` friendly: we give `hyperref` what it needs, but avoid its change to `\@xfootnotemark`.

Any other package that rewrites LaTeX's footnote macros will be incompatible with this package.

```
\appdef\class@documenthook{%
 \@ifxundefined\H@@footnotemark{%
  \let\H@@footnotemark\@footnotemark
 }{}%
 \let\@xfootnotemark\@xfootnotemark@ltx
}%
```

Two thoughts about `hyperref`: what for does it define `\realfootnote`? 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 LaTeX 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!

`\@footnotetext`
`\@mpfootnotetext`
`\@tpfootnotetext`
`\make@footnotetext`
`\set@footnotewidth`

The two procedures `\@footnotetext` and `\@mpfootnotetext` share code. We make that explicit here.

Note that the procedure calling `\make@footnotetext` will open a group with `\bgroup` which is then closed by `\minipagefootnote@drop`.

Difference from LaTeX: here we do not set `\floatingpenalty` to infinity. Doing this must date back to a time when LaTeX could not accomodate split insertions (footnotes). I cannot think of any other reason to do have done this. At any rate, with the `ltxgrid` package, split insertions are specifically 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.

A procedure, `\set@footnotewidth@ii`, illustrates how to do this when in a two-column page grid. 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.

```
\long\def\@footnotetext{%
 \insert\footins\bgroup
  \make@footnotetext
}%
\long\def\@mpfootnotetext{%
 \minipagefootnote@pick
  \make@footnotetext
}%
\def\make@footnotetext#1{%
  \reset@font\footnotesize
  \interlinepenalty\interfootnotelinepenalty
  \splittopskip\footnotesep
  \splitmaxdepth\dp\strutbox
% \floatingpenalty\@MM
  \set@footnotewidth
  \@parboxrestore
```

```
  \protected@edef\@currentlabel{%
   \csname p@footnote\endcsname\@thefnmark
  }%
  \color@begingroup
   \@makefntext{%
    \rule\z@\footnotesep\ignorespaces#1\@finalstrut\strutbox
   }%
  \color@endgroup
 \minipagefootnote@drop
}%
\def\set@footnotewidth{%
 \hsize\columnwidth
 \linewidth\hsize
}%
\def\set@footnotewidth@ii{%
 \hsize\textwidth
 \advance\hsize\columnsep
 \divide\hsize\tw@
 \advance\hsize-\columnsep
 \linewidth\hsize
}%
```

6.10 Floats

6.10.1 Usage notes

We extend the LaTeX 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 unfloated. 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.

```
%\AtBeginDocument{\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.

6.10.2 Robustifying fragile commands

Certain of LaTeX'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`

```
\def\addtocontents#1#2{%
 \protected@write\@auxout{%
  \let \label \@gobble \let \index \@gobble \let \glossary \@gobble
  \def\({\string\(}%
  \def\){\string\)}%
  \def\\{\string\\}%
 }{\string \@writefile {#1}{#2}}%
}%
```

6.10.3 Preparing for the `hyperref` package

`\addcontentsline` `\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 LaTeX. 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.

This means that a document class that uses this package will itself have trouble taking over a ltxutil.dtx.toc file that was written by standard LaTeX. Sigh.

```
\def\addcontentsline#1#2#3{%
 \addtocontents{#1}{%
```

```
  \protect\contentsline{#2}{#3}{\thepage}{}%
 }%
}%
\def\contentsline#1#2#3#4{%
 \csname l@#1\endcsname{#2}{#3}%
}%
\def\label#1{%
 \@bsphack
  \protected@write\@auxout{}{%
   \string\newlabel{#1}{{\@currentlabel}{\thepage}{}{}{}}%
  }%
 \@esphack
}%
```

6.10.4 Footnotes within floats, unfloating floats, float font

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

```
\appdef\class@documenthook{%
 \prepdef\caption{\minipagefootnote@here}%
}%
```

Note on `hyperref` compatability: this change to the `\caption` command is compatible with the "Automated LaTeX 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\@captype
%  \hyper@makecurrent{\@captype}%
%  \@dblarg{\H@caption\@captype}%
%}%
%\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. These
`\minipagefootnote@here` procedures encapsulate all uses of the `\@mpfootins` box.
`\minipagefootnote@foot` Note: `\minipagefootnote@here` must *not* be executed within the MVL!
`\minipagefootnote@pick`
`\minipagefootnote@drop`

```
\def\minipagefootnote@init{%
 \setbox\@mpfootins\box\voidb@x
}%
\def\minipagefootnote@pick{%
 \global\setbox\@mpfootins\vbox\bgroup
  \unvbox\@mpfootins
```

```
}%
\def\minipagefootnote@drop{%
 \egroup
}%
\def\minipagefootnote@here{%
    \par
    \@ifvoid\@mpfootins{}{%
      \vskip\skip\@mpfootins
      \fullinterlineskip
      \@ifinner{%
       \vtop{\unvcopy\@mpfootins}%
       {\setbox\z@\lastbox}%
      }{}%
      \unvbox\@mpfootins
    }%
}%
\def\minipagefootnote@foot{%
 \@ifvoid\@mpfootins{}{%
  \insert\footins\bgroup\unvbox\@mpfootins\egroup
 }%
}%
\def\endminipage{%
    \par
    \unskip
    \minipagefootnote@here
    \@minipagefalse   %% added 24 May 89
  \color@endgroup
  \egroup
  \expandafter\@iiiparbox\@mpargs{\unvbox\@tempboxa}%
}%
```

`\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 state of this Boolean is to be changed by the document class in response to user-selected options. Here we display model code that assigns a default value at `\AtBeginDocument` time.

```
%\AtBeginDocument{%
% \@ifxundefined\floats@sw{\@booleantrue\floats@sw}{}%
%}%
%
```

`\@xfloat` `\@mpmakefntext` The float start-code is redefined to set up footnotes in the style of minipage. Also, 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.

FIXME: why does hyperref override `\@xfootnotenext`?

```
\let\@xfloat@LaTeX\@xfloat
\def\@xfloat#1[#2]{%
  \@xfloat@prep
  \@nameuse{fp@proc@#2}%
  \@ifxundefined\floats@sw{\global\@booleantrue\floats@sw}{}%
  \floats@sw{\@xfloat@LaTeX{#1}[#2]}{\@xfloat@anchored{#1}[]}%
}%
\def\@xfloat@prep{%
```

```
  \let\footnote\footnote@latex
  \def\@mpfn{mpfootnote}%
  \def\thempfn{\thempfootnote}%
% \def\thefootnote{\thempfootnote}%
  \c@mpfootnote\z@
  \let\@footnotetext\@mpfootnotetext
  \let\H@@footnotetext\@mpfootnotetext
  \let\@makefntext\@mpmakefntext
% \samepage
}%
\appdef\class@documenthook{%
 \let\footnote@latex\footnote
}%
%\def\fp@proc@h{\@booleanfalse\floats@sw}%
%\def\fp@proc@H{\@booleanfalse\floats@sw}%
\def\@xfloat@anchored#1[#2]{%
 \def\@captype{#1}%
 \begin@float@pagebreak
%\vbox\bgroup
  \let\end@float\end@float@anchored
  \let\end@dblfloat\end@float@anchored
% do as \@xfloat does:
        \hsize\columnwidth
        \@parboxrestore
        \@floatboxreset
  \minipagefootnote@init
% \pagegrid@col\@ne % Klootch to avoid processing as a float
}%
\def\end@float@anchored{%
  \minipagefootnote@here
  \par\vskip\z@skip      %% \par\vskip\z@ added 15 Dec 87
%\egroup
 \par
 \end@float@pagebreak
}%
\def\begin@float@pagebreak{\par\addvspace\intextsep}%
\def\end@float@pagebreak{\par\addvspace\intextsep}%
\def\@mpmakefntext#1{%
 \parindent=1em
 \noindent
 \hb@xt@1em{\hss\@makefnmark}%
 #1%
}%
```

6.10.5 Writing floats out to a file

`\do@if@floats` The procedure `\do@if@floats` should be executed at `\AtBeginDocument` time, and 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.

```
\def\do@if@floats#1#2{%
 \@ifxundefined\floats@sw{\global\@booleantrue\floats@sw}{}%
 \floats@sw{}{%
```

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

```
\expandafter\newwrite
            \csname#1write\endcsname
\expandafter\def
            \csname#1@stream\endcsname{\jobname#2}%
\expandafter\immediate
\expandafter\openout
            \csname#1write\endcsname
            \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`.

```
\@ifxundefined\@float@LaTeX{%
 \let\@float@LaTeX\@float
 \let\@dblfloat@LaTeX\@dblfloat
 \let\@float\write@float
 \let\@dblfloat\write@floats
}{}%
\let@environment{#1@float}{#1}%
\let@environment{#1@floats}{#1*}%
\@ifxundefined@cs{#1@write}{}{%
 \let@environment{#1}{#1@write}%
}%
}%
```

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

```
\def\triggerpar{\leavevmode\@@par}%
\def\oneapage{\def\begin@float@pagebreak{\newpage}\def\end@float@pagebreak{\newpag
\def\print@float#1#2{%
 \@ifxundefined@cs{#1write}{}{%
  \begingroup
   \@booleanfalse\floats@sw
   #2%
   \raggedbottom
   \def\array@default{v}% floats must
   \let\@float\@float@LaTeX
   \let\@dblfloat\@dblfloat@LaTeX
   \let\trigger@float@par\triggerpar
   \let@environment{#1}{#1@float}%
   \let@environment{#1*}{#1@floats}%
   \expandafter\prepdef\csname#1\endcsname{\trigger@float@par}%
   \expandafter\prepdef\csname#1*\endcsname{\trigger@float@par}%
   \@namedef{fps@#1}{h!}%
   \expandafter\immediate
   \expandafter\closeout
               \csname#1write\endcsname
   \everypar{%
    \global\let\trigger@float@par\relax
    \global\everypar{}\setbox\z@\lastbox
    \@ifxundefined@cs{#1sname}{}{%
     \begin@float@pagebreak
```

```
    \expandafter\section
    \expandafter*%
    \expandafter{%
                 \csname#1sname\endcsname
                }%
   }%
  }%
  \input{\csname#1@stream\endcsname}%
 \endgroup
 \global\expandafter\let\csname#1write\endcsname\relax
}%
}%
```

`\write@float` `\write@floats` `\write@@float`

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

```
\def\write@float#1{\write@@float{#1}{#1}}%
\def\endwrite@float{\@Esphack}%
\def\write@floats#1{\write@@float{#1*}{#1}}%
\def\endwrite@floats{\@Esphack}%
```

`\write@@float`

```
\def\write@@float#1#2{%
 \ifhmode
    \@bsphack
 \fi
 \chardef\@tempc\csname#2write\endcsname
 \toks@{\begin{#1}}%
 \def\@tempb{#1}%
 \expandafter\let\csname end#1\endcsname\endwrite@float
 \catcode`\^^M\active
 \@makeother\{\@makeother\}\@makeother\%
 \write@floatline
}%
```

`\write@floatline` `\@write@floatline` `\float@end@tag`

The procedure `\write@floatline` only parses, and passes its result to `\@write@floatline`, which writes the line to output, then tests the line for the `\end{`<*float*>`}` tokens with aid of the `\float@end@tag` procedure.

```
\begingroup
 \catcode`\[\the\catcode`\{\catcode`\]\the\catcode`\}\@makeother\{\@makeother\}%
 \gdef\float@end@tag#1\end{#2}#3\@nul[%
  \def\@tempa[#2]%
  \@ifx[\@tempa\@tempb][\end[#2]][\write@floatline]%
 ]%
 \obeylines%
 \gdef\write@floatline#1^^M[%
  \begingroup%
   \newlinechar`\^^M%
   \toks@\expandafter[\the\toks@#1]\immediate\write\@tempc[\the\toks@]%
  \endgroup%
  \toks@[]%
  \float@end@tag#1\end{}\@nul%
 ]%
\endgroup
```

6.11 Counters

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

```
\def\@alph#1{\ifcase#1\or a\or b\or c\or d\else\@ialph{#1}\fi}
\def\@ialph#1{\ifcase#1\or \or \or \or \or e\or f\or g\or h\or i\or j\or
  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
  y\or z\or aa\or bb\or cc\or dd\or ee\or ff\or gg\or hh\or ii\or jj\or
  kk\or ll\or mm\or nn\or oo\or pp\or qq\or rr\or ss\or tt\or uu\or
  vv\or ww\or xx\or yy\or zz\else\@ctrerr\fi}
```

6.12 Customization of Sections

Patch the standard LaTeX 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 `\@seccntformat`.
- 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.

```
\def\@startsection#1#2#3#4#5#6{%
 \@startsection@hook
 \if@noskipsec \leavevmode \fi
 \par
 \@tempskipa #4\relax
 \@afterindenttrue
 \ifdim \@tempskipa <\z@
  \@tempskipa -\@tempskipa \@afterindentfalse
 \fi
 \if@nobreak
  \everypar{}%
 \else
  \addpenalty\@secpenalty\addvspace\@tempskipa
 \fi
 \@ifstar
  {\@dblarg{\@ssect@ltx{#1}{#2}{#3}{#4}{#5}{#6}}}%
  {\@dblarg{\@sect@ltx {#1}{#2}{#3}{#4}{#5}{#6}}}%
```

```
}%
\def\@startsection@hook{}%
```

\@sect When defining \@svsec, do not expand \@seccntformat. 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 \@hangfrom and \@runin@to can be used to process the argument of the head. The head can define, e.g., \@hangfrom@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 LaTeX procedure.

```
\class@info
  {Repairing broken LateX \string\@sect}%
\def\@sect@ltx#1#2#3#4#5#6[#7]#8{%
  \@ifnum{#2>\c@secnumdepth}{%
    \def\H@svsec{\phantomsection}%
    \let\@svsec\@empty
  }{%
    \H@refstepcounter{#1}%
    \def\H@svsec{%
     \phantomsection
    }%
    \protected@edef\@svsec{{#1}}%
    \@ifundefined{@#1cntformat}{%
     \prepdef\@svsec\@seccntformat
    }{%
     \expandafter\prepdef
     \expandafter\@svsec
                 \csname @#1cntformat\endcsname
    }%
  }%
  \@tempskipa #5\relax
  \@ifdim{\@tempskipa>\z@}{%
    \begingroup
      \interlinepenalty \@M
      #6{%
       \@ifundefined{@hangfrom@#1}{\@hang@from}{\csname @hangfrom@#1\endcsname}%
       {\hskip#3\relax\H@svsec}{\@svsec}{#8}%
      }%
      \@@par
    \endgroup
    \@ifundefined{#1mark}{\@gobble}{\csname #1mark\endcsname}{#7}%
    \addcontentsline{toc}{#1}{%
      \@ifnum{#2>\c@secnumdepth}{%
       \protect\numberline{}%
      }{%
       \protect\numberline{\csname the#1\endcsname}%
      }%
      #8}%
  }{%
    \def\@svsechd{%
```

```
      #6{%
       \@ifundefined{@runin@to@#1}{\@runin@to}{\csname @runin@to@#1\endcsname}%
       {\hskip#3\relax\H@svsec}{\@svsec}{#8}%
      }%
      \@ifundefined{#1mark}{\@gobble}{\csname #1mark\endcsname}{#7}%
      \addcontentsline{toc}{#1}{%
        \@ifnum{#2>\c@secnumdepth}{%
         \protect\numberline{}%
        }{%
         \protect\numberline{\csname the#1\endcsname}%
        }%
        #8}%
    }%
  }%
  \@xsect{#5}%
}%
\def\@hang@from#1#2#3{\@hangfrom{#1#2}#3}%
\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 compatability 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`.

```
\def\@ssect@ltx#1#2#3#4#5#6[#7]#8{%
% \def\@currentlabelname{#8}%
  \def\H@svsec{\phantomsection}%
  \@tempskipa #5\relax
  \@ifdim{\@tempskipa>\z@}{%
    \begingroup
      \interlinepenalty \@M
      #6{%
       \@ifundefined{@hangfroms@#1}{\@hang@froms}{\csname @hangfroms@#1\endcsname}
%      {\hskip#3\relax\H@svsec}{\Sectionformat{#8}{#1}}%
       {\hskip#3\relax\H@svsec}{#8}%
      }%
      \@@par
    \endgroup
    \@ifundefined{#1smark}{\@gobble}{\csname #1smark\endcsname}{#7}%
    \addcontentsline{toc}{#1}{\protect\numberline{}#8}%
  }{%
    \def\@svsechd{%
      #6{%
       \@ifundefined{@runin@tos@#1}{\@runin@tos}{\csname @runin@tos@#1\endcsname}%
%      {\hskip#3\relax\H@svsec}{\Sectionformat{#8}{#1}}%
       {\hskip#3\relax\H@svsec}{#8}%
      }%
      \@ifundefined{#1smark}{\@gobble}{\csname #1smark\endcsname}{#7}%
      \addcontentsline{toc}{#1}{\protect\numberline{}#8}%
```

```
    }%
  }%
  \@xsect{#5}%
}%
\def\@hang@froms#1#2{#1#2}%
\def\@runin@tos #1#2{#1#2}%
```

\init@documenthook

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 ltxutil.dtx.aux file will now 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.

```
\appdef\init@documenthook{%
 \providecommand\phantomsection{}%
%\@ifx{\Sectionformat\@undefined}{\let\Sectionformat\@firstoftwo}{}%
 \providecommand\hyper@anchor[1]{}%
 \providecommand\hyper@last{}%
 \providecommand\Hy@raisedlink[1]{#1}%
 \providecommand\hyper@anchorstart[1]{}%
 \providecommand\hyper@anchorend{}%
 \providecommand\hyper@linkstart[2]{}%
 \providecommand\hyper@linkend{}%
}%
\let\H@refstepcounter\refstepcounter
```

\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 LaTeX default.

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

6.13 Patch the tabular and array Environments

\endtabular
\endarray

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 LaTeX array and tabular environments. I know of none that do so.

```
\appdef\class@documenthook{%
```

```
\@ifpackageloaded{array}{\switch@array}{\switch@tabular}%
\prepdef\endtabular{\endtabular@hook}%
\@provide\endtabular@hook{}%
\prepdef\endarray{\endarray@hook}%
\@provide\endarray@hook{}%
\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.

```
\prepdef\@tabular{\tabular@hook}%
\@provide\tabular@hook{}%
}%
```

`\switch@tabular` `\switch@array`

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

```
\def\switch@tabular{%
\let\@array@sw\@array@sw@array
\@ifx{\@array\@array@LaTeX}{%
 \@ifx{\multicolumn\multicolumn@LaTeX}{%
  \@ifx{\@tabular\@tabular@LaTeX}{%
   \@ifx{\@tabarray\@tabarray@LaTeX}{%
    \@ifx{\array\array@LaTeX}{%
     \@ifx{\endarray\endarray@LaTeX}{%
      \@ifx{\endtabular\endtabular@LaTeX}{%
       \@ifx{\@mkpream\@mkpream@LaTeX}{%
        \@ifx{\@addamp\@addamp@LaTeX}{%
         \@ifx{\@arrayacol\@arrayacol@LaTeX}{%
          \@ifx{\@tabacol\@tabacol@LaTeX}{%
           \@ifx{\@arrayclassz\@arrayclassz@LaTeX}{%
            \@ifx{\@tabclassiv\@tabclassiv@LaTeX}{%
             \@ifx{\@arrayclassiv\@arrayclassiv@LaTeX}{%
              \@ifx{\@tabclassz\@tabclassz@LaTeX}{%
               \@ifx{\@classv\@classv@LaTeX}{%
                \@ifx{\hline\hline@LaTeX}{%
                 \@ifx{\@tabularcr\@tabularcr@LaTeX}{%
                  \@ifx{\@xtabularcr\@xtabularcr@LaTeX}{%
                   \@ifx{\@xargarraycr\@xargarraycr@LaTeX}{%
                    \@ifx{\@yargarraycr\@yargarraycr@LaTeX}{%
                     \true@sw
                    }{%
                     \false@sw
                    }%
                   }{%
                    \false@sw
                   }%
                  }{%
                   \false@sw
                  }%
                 }{%
                  \false@sw
                 }%
                }{%
                 \false@sw
```

```
                  }%
                 }{%
                  \false@sw
                 }%
                }{%
                 \false@sw
                }%
               }{%
                \false@sw
               }%
              }{%
               \false@sw
              }%
             }{%
              \false@sw
             }%
            }{%
             \false@sw
            }%
           }{%
            \false@sw
           }%
          }{%
           \false@sw
          }%
         }{%
          \false@sw
         }%
        }{%
         \false@sw
        }%
       }{%
        \false@sw
       }%
      }{%
       \false@sw
      }%
     }{%
      \false@sw
     }%
    }{%
     \false@sw
    }%
   }{%
    \false@sw
   }%
  }{%
   \false@sw
  }%
}{%
 \false@sw
}%
{%
 \class@info{Patching LaTeX tabular.}%
}{%
 \class@info{Unrecognized LaTeX tabular. Please update this document class! (Proc
}%
```

```
\let\@array\@array@ltx
\let\multicolumn\multicolumn@ltx
\let\@tabular\@tabular@ltx
\let\@tabarray\@tabarray@ltx
\let\array\array@ltx
\let\endarray\endarray@ltx
\let\endtabular\endtabular@ltx
\let\@mkpream\@mkpream@ltx
\let\@addamp\@addamp@ltx
\let\@arrayacol\@arrayacol@ltx
\let\@tabacol\@tabacol@ltx
\let\@arrayclassz\@arrayclassz@ltx
\let\@tabclassiv\@tabclassiv@ltx
\let\@arrayclassiv\@arrayclassiv@ltx
\let\@tabclassz\@tabclassz@ltx
\let\@classv\@classv@ltx
\let\hline\hline@ltx
\let\@tabularcr\@tabularcr@ltx
\let\@xtabularcr\@xtabularcr@ltx
\let\@xargarraycr\@xargarraycr@ltx
\let\@yargarraycr\@yargarraycr@ltx
}%
\def\switch@array{%
\let\@array@sw\@array@sw@LaTeX
\@ifx{\@array\@array@array}{%
 \@ifx{\@tabular\@tabular@array}{%
  \@ifx{\@tabarray\@tabarray@array}{%
   \@ifx{\array\array@array}{%
    \@ifx{\endarray\endarray@array}{%
     \@ifx{\endtabular\endtabular@array}{%
      \@ifx{\@mkpream\@mkpream@array}{%
       \@ifx{\@classx\@classx@array}{%
        \@ifx{\insert@column\insert@column@array}{%
         \@ifx{\@arraycr\@arraycr@array}{%
          \@ifx{\@xarraycr\@xarraycr@array}{%
           \@ifx{\@xargarraycr\@xargarraycr@array}{%
            \@ifx{\@yargarraycr\@yargarraycr@array}{%
             \true@sw
        }{%
         \false@sw
        }%
        }{%
         \false@sw
        }%
        }{%
         \false@sw
        }%
        }{%
         \false@sw
        }%
        }{%
         \false@sw
        }%
       }{%
```

```
          \false@sw
         }%
        }{%
         \false@sw
        }%
       }{%
        \false@sw
       }%
      }{%
       \false@sw
      }%
     }{%
      \false@sw
     }%
    }{%
     \false@sw
    }%
   }{%
    \false@sw
   }%
  }{%
   \false@sw
  }%
 }{%
  \false@sw
 }{%
  \class@info{Patching array package.}%
 }{%
  \class@info{Unrecognized array package. Please update this document class! (Proc
 }%
 \let\@array    \@array@array@new
 \let\@@array   \@array % Cosi fan tutti
 \let\@tabular  \@tabular@array@new
 \let\@tabarray \@tabarray@array@new
 \let\array     \array@array@new
 \let\endarray  \endarray@array@new
 \let\endtabular\endtabular@array@new
 \let\@mkpream  \@mkpream@array@new
 \let\@classx   \@classx@array@new
 \let\@arrayacol\@arrayacol@ltx
 \let\@tabacol  \@tabacol@ltx
 \let\insert@column\insert@column@array@new
 \expandafter\let\csname endtabular*\endcsname\endtabular % Cosi fan tutti
 \let\@arraycr  \@arraycr@new
 \let\@xarraycr \@xarraycr@new
 \let\@xargarraycr\@xargarraycr@new
 \let\@yargarraycr\@yargarraycr@new
}%
```

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

```
\def\@array@sw@LaTeX{\@ifx{\\\@tabularcr}}%
\def\@array@sw@array{\@ifx{\d@llarbegin\begingroup}}%
```

`\@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).

```
\def\@tabular@LaTeX{%
 \leavevmode
 \hbox\bgroup$%
  \let\@acol\@tabacol
  \let\@classz\@tabclassz
  \let\@classiv\@tabclassiv
  \let\\\@tabularcr
  \@tabarray
}%
\def\@tabular@ltx{%
  \let\@acoll\@tabacoll
  \let\@acolr\@tabacolr
  \let\@acol\@tabacol
  \let\@classz\@tabclassz
  \let\@classiv\@tabclassiv
  \let\\\@tabularcr
  \@tabarray
}%
\def\@tabular@array{%
 \leavevmode
 \hbox\bgroup$%
  \col@sep\tabcolsep
  \let\d@llarbegin\begingroup
  \let\d@llarend\endgroup
  \@tabarray
}%
\def\@tabular@array@new{%
  \let\@acoll\@tabacoll
  \let\@acolr\@tabacolr
  \let\@acol\@tabacol
  \let\col@sep\@undefined
  \let\d@llarbegin\begingroup
  \let\d@llarend\endgroup
  \@tabarray
}%
```

`\@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 LaTeX. Under some circumstances, we want to change this to, say, 'v'.

FIXME: must decouple `array` and `tabular`.

```
\def\@tabarray@LaTeX{%
 \m@th\@ifnextchar[\@array{\@array[c]}%
}%
\def\@tabarray@ltx{%
 \m@th\@ifnextchar[\@array{\expandafter\@array\expandafter[\array@default]}%
}%
\def\@tabarray@array{%
 \@ifnextchar[{\@@array}{\@@array[c]}%
}%
\def\@tabarray@array@new{%
 \@ifnextchar[{\@@array}{\expandafter\@@array\expandafter[\array@default]}%
}%
```

`\@tabularcr` `\@tbpen` `\@tabularcr` `\@xtabularcr` `\@xargarraycr` `\@yargarraycr` `\@arraycr` `\@xarraycr` We provide for the `\\` command within `tabular` to provide control over page breaking,

just the same as that of `eqnarray`.

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.

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 `\@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 `\@eqncr`: it sets `\@tbpen` to `\@M` if the star was given.

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

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

```
\newcount\intertabularlinepenalty
\intertabularlinepenalty=100
\newcount\@tbpen
\appdef\samepage{\intertabularlinepenalty\@M}%
\def\@tabularcr@LaTeX{{\ifnum 0=‘}\fi \@ifstar \@xtabularcr \@xtabularcr}%
\def\@tabularcr@ltx{{\ifnum 0=‘}\fi \@ifstar {\global \@tbpen \@M \@xtabularcr }{\
\def\@xtabularcr@LaTeX{\@ifnextchar [\@argtabularcr {\ifnum 0=‘{\fi }\cr }}%
\def\@xtabularcr@ltx{\@ifnextchar [\@argtabularcr {\ifnum 0=‘{\fi }\cr \noalign {\
\def\@xargarraycr@LaTeX#1{\@tempdima #1\advance \@tempdima \dp \@arstrutbox \vrule
\def\@xargarraycr@ltx#1{\@tempdima #1\advance \@tempdima \dp \@arstrutbox \vrule \
\def\@yargarraycr@LaTeX#1{\cr \noalign {\vskip #1}}%
\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.

```
\def\@arraycr@array{%
 \relax
 \iffalse{\fi\ifnum 0=‘}\fi
 \@ifstar \@xarraycr \@xarraycr
}%
\def\@arraycr@new{%
 \relax
 \iffalse{\fi\ifnum 0=‘}\fi
 \@ifstar {\global \@tbpen \@M \@xarraycr }{\global \@tbpen \intertabularlinepenal
}%
\def\@xarraycr@array{%
 \@ifnextchar [%]
 \@argarraycr {\ifnum 0=‘{}\fi\cr}%
}%
\def\@xarraycr@new{%
 \@ifnextchar [%]
 \@argarraycr {\ifnum 0=‘{}\fi\cr \noalign {\penalty \@tbpen }}%
}%
\def\@xargarraycr@array#1{%
 \unskip
 \@tempdima #1\advance\@tempdima \dp\@arstrutbox
 \vrule \@depth\@tempdima \@width\z@
 \cr
```

```
}%
\def\@xargarraycr@new#1{%
 \unskip
 \@tempdima #1\advance\@tempdima \dp\@arstrutbox
 \vrule \@depth\@tempdima \@width\z@
 \cr
 \noalign {\penalty \@tbpen }%
}%
\def\@yargarraycr@array#1{%
 \cr
 \noalign{\vskip #1}%
}%
\def\@yargarraycr@new#1{%
 \cr
 \noalign{\penalty \@tbpen \vskip #1}%
}%
```

`\array` We provide old and new versions of the `\array` procedure for both LaTeX 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.

```
\def\array@LaTeX{%
 \let\@acol\@arrayacol
 \let\@classz\@arrayclassz
 \let\@classiv\@arrayclassiv
 \let\\\@arraycr
 \let\@halignto\@empty
 \@tabarray
}%
\def\array@ltx{%
 \@ifmmode{}{\@badmath$}%
 \let\@acoll\@arrayacol
 \let\@acolr\@arrayacol
 \let\@acol\@arrayacol
 \let\@classz\@arrayclassz
 \let\@classiv\@arrayclassiv
 \let\\\@arraycr
 \let\@halignto\@empty
 \@tabarray
}%
\def\array@array{%
 \col@sep\arraycolsep
 \def\d@llarbegin{$}\let\d@llarend\d@llarbegin\gdef\@halignto{}%
 \@tabarray
}
\def\array@array@new{%
 \@ifmmode{}{\@badmath$}%
 \let\@acoll\@arrayacol
 \let\@acolr\@arrayacol
 \let\@acol\@arrayacol
\let\col@sep\@undefined
 \def\d@llarbegin{$}%
 \let\d@llarend\d@llarbegin
 \gdef\@halignto{}%
```

```
 \@tabarray
}%
```

\@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.

```
\def\@array@LaTeX[#1]#2{%
  \if #1t\vtop \else \if#1b\vbox \else \vcenter \fi\fi
  \bgroup
  \setbox\@arstrutbox\hbox{%
    \vrule \@height\arraystretch\ht\strutbox
           \@depth\arraystretch \dp\strutbox
           \@width\z@}%
  \@mkpream{#2}%
  \edef\@preamble{%
    \ialign \noexpand\@halignto
      \bgroup \@arstrut \@preamble \tabskip\z@skip \cr}%
  \let\@startpbox\@@startpbox \let\@endpbox\@@endpbox
  \let\tabularnewline\\%
    \let\par\@empty
    \let\@sharp##%
    \set@typeset@protect
    \lineskip\z@skip\baselineskip\z@skip
    \ifhmode \@preamerr\z@ \@@par\fi
    \@preamble
}%
\def\@array@ltx[#1]#2{%
 \@nameuse{@array@align@#1}%
  \set@arstrutbox
  \@mkpream{#2}%
  \prepdef\@preamble{%
    \tabskip\tabmid@skip
    \@arstrut
  }%
  \appdef\@preamble{%
    \tabskip\tabright@skip
    \cr
    \array@row@pre
  }%
% \let\@startpbox\@@startpbox
% \let\@endpbox\@@endpbox
  \let\tabularnewline\\%
  \let\par\@empty
  \let\@sharp##%
  \set@typeset@protect
  \lineskip\z@skip\baselineskip\z@skip
  \tabskip\tableft@skip\relax
  \ifhmode \@preamerr\z@ \@@par\fi
```

```
  \everycr{}%
  \expandafter\halign\expandafter\@halignto\expandafter\bgroup\@preamble
}%
%
\def\set@arstrutbox{%
  \setbox\@arstrutbox\hbox{%
    \vrule \@height\arraystretch\ht\strutbox
           \@depth\arraystretch \dp\strutbox
           \@width\z@
  }%
}%
```

\@array@array

```
\def\@array@array[#1]#2{%
  \@tempdima \ht \strutbox
  \advance \@tempdima by\extrarowheight
  \setbox \@arstrutbox \hbox{\vrule
             \@height \arraystretch \@tempdima
             \@depth \arraystretch \dp \strutbox
             \@width \z@}%
  \begingroup
  \@mkpream{#2}%
  \xdef\@preamble{\noexpand \ialign \@halignto
                  \bgroup \@arstrut \@preamble
                          \tabskip \z@ \cr}%
  \endgroup
  \@arrayleft
  \if #1t\vtop \else \if#1b\vbox \else \vcenter \fi \fi
  \bgroup
  \let \@sharp ##\let \protect \relax
  \lineskip \z@
  \baselineskip \z@
  \m@th
  \let\\\@arraycr \let\tabularnewline\\\let\par\@empty \@preamble
}%
\def\@array@array@new[#1]#2{%
  \@tempdima\ht\strutbox
  \advance\@tempdima by\extrarowheight
  \setbox\@arstrutbox\hbox{%
   \vrule \@height\arraystretch\@tempdima
          \@depth \arraystretch\dp\strutbox
          \@width \z@
  }%
  \begingroup
   \@mkpream{#2}%
   \xdef\@preamble{\@preamble}%
  \endgroup
  \prepdef\@preamble{%
   \tabskip\tabmid@skip
    \@arstrut
  }%
  \appdef\@preamble{%
   \tabskip\tabright@skip
   \cr
```

```
  \array@row@pre
 }%
 \@arrayleft
 \@nameuse{@array@align@#1}%
 \m@th
 \let\\\@arraycr
 \let\tabularnewline\\%
 \let\par\@empty
 \let\@sharp##%
 \set@typeset@protect
 \lineskip\z@\baselineskip\z@
 \tabskip\tableft@skip
 \everycr{}%
 \expandafter\halign\expandafter\@halignto\expandafter\bgroup\@preamble
}%
```

\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.

```
\def\endarray@LaTeX{%
 \crcr\egroup\egroup
}%
\def\endarray@ltx{%
 \crcr\array@row@pst\egroup\egroup
}%
\def\endarray@array{%
 \crcr \egroup \egroup \@arrayright \gdef\@preamble{}%
}%
\def\endarray@array@new{%
 \crcr\array@row@pst\egroup\egroup % Same as \endarray@ltx
 \@arrayright
 \global\let\@preamble\@empty
}%
```

\endtabular

```
\def\endtabular@LaTeX{%
 \crcr\egroup\egroup $\egroup
}%
\def\endtabular@ltx{%
 \endarray
}%
\def\endtabular@array{%
 \endarray $\egroup
}%
\def\endtabular@array@new{%
 \endarray
}%
```

endtabular* Here we provide a proper definition for the star-form of \end{endtabular}. It is one of the enduring curiosities that the LaTeX kernel continues to use dangerously and inappropriately "optimized" definitions for such commands.

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

\multicolumn

```
\long\def\multicolumn@LaTeX#1#2#3{%
 \multispan{#1}\begingroup
  \@mkpream{#2}%
  \def\@sharp{#3}\set@typeset@protect
  \let\@startpbox\@@startpbox\let\@endpbox\@@endpbox
  \@arstrut \@preamble\hbox{}\endgroup\ignorespaces
}%
\long\def\multicolumn@ltx#1#2#3{%
 \multispan{#1}%
 \begingroup
  \@mkpream{#2}%
  \def\@sharp{#3}%
  \set@typeset@protect
 %\let\@startpbox\@@startpbox\let\@endpbox\@@endpbox
  \@arstrut
  \@preamble
  \hbox{}%
 \endgroup
 \ignorespaces
}%
```

\@array@align@
\array@default

Here are the various procedures for the vertical alignment options. The change from standard LaTeX 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.

```
\def\@array@align@t{\leavevmode\vtop\bgroup}%
\def\@array@align@b{\leavevmode\vbox\bgroup}%
\def\@array@align@c{\leavevmode\@ifmmode{\vcenter\bgroup}{$\vcenter\bgroup\aftergr
\def\@array@align@v{%
 \@ifmmode{%
  \@badmath
  \vcenter\bgroup
 }{%
  \@ifinner{%
   $\vcenter\bgroup\aftergroup$
  }{%
   \@@par\bgroup
  }%
 }%
}%
\def\array@default{c}%
```

\array@row@pre
\array@row@pst
\array@row@rst

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.

```
\def\array@row@rst{%
 \let\@array@align@v\@array@align@c
}%
```

```
\def\array@row@pre{}%
\def\array@row@pst{}%
```

\toprule
\colrule
\botrule

Default definitions for \toprule, \colrule, \botrule

```
\newcommand\toprule{\tab@rule{\column@font}{\column@fil}{\frstrut}}%
\newcommand\colrule{\unskip\lrstrut\\\tab@rule{\body@font}{}{\frstrut}}%
\newcommand\botrule{\unskip\lrstrut\\\noalign{\hline@rule}{}}%
```

\hline

```
\def\hline@LaTeX{%
 \noalign{\ifnum0=‘}\fi\hrule \@height \arrayrulewidth \futurelet
   \reserved@a\@xhline
}%
\def\hline@ltx{%
 \noalign{%
  \ifnum0=‘}\fi
  \hline@rule
  \futurelet\reserved@a\@xhline
 % \noalign ended in \@xhline
}%
\def\@xhline@unneeded{%
 \say\reserved@a
 \ifx\reserved@a\hline
  \vskip\doublerulesep
  \vskip-\arrayrulewidth
 \fi
 \ifnum0=‘{\fi}%
}%
\def\tab@rule#1#2#3{%
 \crcr
 \noalign{%
  \hline@rule
  \gdef\@arstrut@hook{%
   \global\let\@arstrut@hook\@empty
   #3%
  }%
  \gdef\cell@font{#1}%
  \gdef\cell@fil{#2}%
 }%
}%
\def\column@font{}%
\def\column@fil{}%
\def\body@font{}%
\def\cell@font{}%
\def\frstrut{}%
\def\lrstrut{}%
```

\@arstrut@hline
\@arstrut@org
\@arstrut@hook
\@arstrutbox@hline
\set@arstrutbox
\hline@rule

The procedure \@arstrut@hline is substantially the same as \@arstrut, except the strut copied in is \@arstrutbox@hlineinstead of \@arstrutbox.

The procedure \@arstrut@hook is redefined in \tab@rule!

The register \@arstrutbox@hline.

We append to \set@arstrutbox the code necessary to set a strut following an \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 klootch, a magic number.

```
\def\@arstrut@hline{%
 \relax
 \@ifmmode{\copy}{\unhcopy}\@arstrutbox@hline
 \@arstrut@hook
}%
%
\let\@arstrut@org\@arstrut
\def\@arstrut@hook{%
 \global\let\@arstrut\@arstrut@org
}%
%
\newbox\@arstrutbox@hline
\appdef\set@arstrutbox{%
  \setbox\@arstrutbox@hline\hbox{%
    \setbox\z@\hbox{$0^{0}_{}$}%
    \dimen@\ht\z@\advance\dimen@\@arstrut@hline@clnc
    \@ifdim{\dimen@<\arraystretch\ht\strutbox}{\dimen@=\arraystretch\ht\strutbox}{
    \vrule \@height\dimen@
           \@depth\arraystretch \dp\strutbox
           \@width\z@
  }%
}%
%
\def\hline@rule{%
 \hrule \@height \arrayrulewidth
 \global\let\@arstrut\@arstrut@hline
}%
\def\@arstrut@hline@clnc{2\p@}% % Klootch: magic number
```

`\tableft@skip`

```
\def\tableft@skip{\z@skip}%
\def\tabmid@skip{\z@skip}%\@flushglue
\def\tabright@skip{\z@skip}%
\def\tableftsep{\tabcolsep}%
\def\tabmidsep{\tabcolsep}%
\def\tabrightsep{\tabcolsep}%
\def\cell@fil{}%
\def\pbox@hook{}%
```

`\@arstrut`

```
\appdef\@arstrut{\@arstrut@hook}%
\let\@arstrut@hook\@empty
\def\@addtopreamble{\appdef\@preamble}%
```

`\@mkpream`

```
\def\@mkpream@LaTeX#1{%
  \@firstamptrue\@lastchclass6
  \let\@preamble\@empty
  \let\protect\@unexpandable@protect
  \let\@sharp\relax
```

```
  \let\@startpbox\relax\let\@endpbox\relax
  \@expast{#1}%
  \expandafter\@tfor \expandafter
    \@nextchar \expandafter:\expandafter=\reserved@a\do
       {\@testpach\@nextchar
    \ifcase \@chclass \@classz \or \@classi \or \@classii \or \@classiii
      \or \@classiv \or\@classv \fi\@lastchclass\@chclass}%
  \ifcase \@lastchclass \@acol
      \or \or \@preamerr \@ne\or \@preamerr \tw@\or \or \@acol \fi
}%
\def\@mkpream@ltx#1{%
 \@firstamptrue
 \@lastchclass6
 \let\@preamble\@empty
 \let\protect\@unexpandable@protect
 \let\@sharp\relax
%\let\@startpbox\relax\let\@endpbox\relax
 \@expast{#1}%
 \expandafter\@tfor\expandafter\@nextchar\expandafter:\expandafter=\reserved@a
 \do{%
  \expandafter\@testpach\expandafter{\@nextchar}%
  \ifcase\@chclass
   \@classz
  \or
   \@classi
  \or
   \@classii
  \or
   \@classiii
  \or
   \@classiv
  \or
   \@classv
  \fi
  \@lastchclass\@chclass
 }%
 \ifcase\@lastchclass
  \@acolr % right-hand column
 \or
 \or
  \@preamerr\@ne
 \or
  \@preamerr\tw@
 \or
 \or
  \@acolr % right-hand column
 \fi
}%
```

`\insert@column`

```
\def\insert@column@array{%
   \the@toks \the \@tempcnta
   \ignorespaces \@sharp \unskip
   \the@toks \the \count@ \relax
```

```
}%
\def\insert@column@array@new{%
 \the@toks\the\@tempcnta
 \array@row@rst\cell@font
 \ignorespaces\@sharp\unskip
 \the@toks\the\count@
 \relax
}%
```

`\@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`.

```
\def\@mkpream@relax{%
 \let\tableftsep\relax
 \let\tabmidsep\relax
 \let\tabrightsep\relax
 \let\array@row@rst\relax
 \let\cell@font\relax
 \let\@startpbox\relax
}%
```

`\@mkpream`

```
\def\@mkpream@array#1{%
   \gdef\@preamble{}\@lastchclass 4 \@firstamptrue
   \let\@sharp\relax \let\@startpbox\relax \let\@endpbox\relax
   \@temptokena{#1}\@tempswatrue
   \@whilesw\if@tempswa\fi{\@tempswafalse\the\NC@list}%
   \count@\m@ne
   \let\the@toks\relax
   \prepnext@tok
   \expandafter \@tfor \expandafter \@nextchar
    \expandafter :\expandafter =\the\@temptokena \do
   {\@testpach
   \ifcase \@chclass \@classz \or \@classi \or \@classii
     \or \save@decl \or \or \@classv \or \@classvi
     \or \@classvii \or \@classviii
     \or \@classx
     \or \@classx \fi
   \@lastchclass\@chclass}%
   \ifcase\@lastchclass
   \@acol \or
   \or
   \@acol \or
   \@preamerr \thr@@ \or
   \@preamerr \tw@ \@addtopreamble\@sharp \or
   \or
```

```
   \else  \@preamerr \@ne \fi
   \def\the@toks{\the\toks}%
}%
\def\@mkpream@array@new#1{%
 \gdef\@preamble{}%
 \@lastchclass\f@ur
 \@firstamptrue
 \let\@sharp\relax
 \@mkpream@relax
%\let\@startpbox\relax\let\@endpbox\relax
 \@temptokena{#1}\@tempswatrue
 \@whilesw\if@tempswa\fi{\@tempswafalse\the\NC@list}%
 \count@\m@ne
 \let\the@toks\relax
 \prepnext@tok
 \expandafter\@tfor\expandafter\@nextchar\expandafter:\expandafter=\the\@temptoken
 \do{%
  \@testpach
  \ifcase\@chclass
   \@classz
  \or
   \@classi
  \or
   \@classii
  \or
   \save@decl
  \or
  \or
   \@classv
  \or
   \@classvi
  \or
   \@classvii
  \or
   \@classviii
  \or
   \@classx
  \or
   \@classx
  \fi
  \@lastchclass\@chclass
 }%
 \ifcase\@lastchclass
  \@acolr % right-hand column
 \or
 \or
  \@acolr % right-hand column
 \or
  \@preamerr\thr@@
 \or
  \@preamerr\tw@\@addtopreamble\@sharp
 \or
 \or
 \else
```

```
  \@preamerr\@ne
 \fi
 \def\the@toks{\the\toks}%
}%
```

`\@addamp`

```
\def\@addamp@LaTeX{%
  \if@firstamp\@firstampfalse\else\edef\@preamble{\@preamble &}\fi
}%
\def\@addamp@ltx{%
 \if@firstamp\@firstampfalse\else\@addtopreamble{&}\fi
}%
```

`\@arrayacol`

```
\def\@arrayacol@LaTeX{%
 \edef\@preamble{\@preamble \hskip \arraycolsep}%
}%
\def\@arrayacol@ltx{%
 \@addtopreamble{\hskip\arraycolsep}%
}%
```

`\@tabacol`

```
\def\@tabacoll{%
 \@addtopreamble{\hskip\tableftsep\relax}%
}%
\def\@tabacol@LaTeX{%
 \edef\@preamble{\@preamble \hskip \tabcolsep}%
}%
\def\@tabacol@ltx{%
 \@addtopreamble{\hskip\tabmidsep\relax}%
}%
\def\@tabacolr{%
 \@addtopreamble{\hskip\tabrightsep\relax}%
}%
```

`\@arrayclassz`

```
\def\@arrayclassz@LaTeX{%
 \ifcase \@lastchclass \@acolampacol \or \@ampacol \or
   \or \or \@addamp \or
   \@acolampacol \or \@firstampfalse \@acol \fi
 \edef\@preamble{\@preamble
  \ifcase \@chnum
     \hfil$\relax\@sharp$\hfil \or $\relax\@sharp$\hfil
    \or \hfil$\relax\@sharp$\fi}%
}%
\def\@arrayclassz@ltx{%
 \ifcase\@lastchclass
  \@acolampacol
 \or
  \@ampacol
 \or
 \or
 \or
  \@addamp
```

```
 \or
  \@acolampacol
 \or
  \@firstampfalse\@acoll
 \fi
 \ifcase\@chnum
  \@addtopreamble{%
   \hfil\array@row@rst$\relax\@sharp$\hfil
  }%
 \or
  \@addtopreamble{%
   \array@row@rst$\relax\@sharp$\hfil
  }%
 \or
  \@addtopreamble{%
   \hfil\array@row@rst$\relax\@sharp$%
  }%
 \fi
}%
```

\@tabclassz

```
\def\@tabclassz@LaTeX{%
  \ifcase\@lastchclass
    \@acolampacol
  \or
    \@ampacol
  \or
  \or
  \or
    \@addamp
  \or
    \@acolampacol
  \or
    \@firstampfalse\@acol
  \fi
  \edef\@preamble{%
    \@preamble{%
      \ifcase\@chnum
        \hfil\ignorespaces\@sharp\unskip\hfil
      \or
        \hskip1sp\ignorespaces\@sharp\unskip\hfil
      \or
        \hfil\hskip1sp\ignorespaces\@sharp\unskip
      \fi}}%
}%
\def\@tabclassz@ltx{%
 \ifcase\@lastchclass
  \@acolampacol
 \or
  \@ampacol
 \or
 \or
 \or
  \@addamp
```

```
 \or
  \@acolampacol
 \or
  \@firstampfalse\@acoll
 \fi
 \ifcase\@chnum
  \@addtopreamble{%
   {\hfil\array@row@rst\cell@font\ignorespaces\@sharp\unskip\hfil}%
  }%
 \or
  \@addtopreamble{%
   {\cell@fil\hskip1sp\array@row@rst\cell@font\ignorespaces\@sharp\unskip\hfil}%
  }%
 \or
  \@addtopreamble{%
   {\hfil\hskip1sp\array@row@rst\cell@font\ignorespaces\@sharp\unskip\cell@fil}%
  }%
 \fi
}%
```

\@tabclassiv

```
\def\@tabclassiv@LaTeX{%
 \@addtopreamble\@nextchar
}%
\def\@tabclassiv@ltx{%
 \expandafter\@addtopreamble\expandafter{\@nextchar}%
}%
```

\@arrayclassiv

```
\def\@arrayclassiv@LaTeX{%
 \@addtopreamble{$\@nextchar$}%
}%
\def\@arrayclassiv@ltx{%
 \expandafter\@addtopreamble\expandafter{\expandafter$\@nextchar$}%
}%
```

\@classv

```
\def\@classv@LaTeX{%
 \@addtopreamble{\@startpbox{\@nextchar}\ignorespaces
 \@sharp\@endpbox}%
}%
\def\@classv@ltx{%
 \expandafter\@addtopreamble
 \expandafter{%
 \expandafter \@startpbox
 \expandafter {\@nextchar}%
 \pbox@hook\array@row@rst\cell@font\ignorespaces\@sharp\@endpbox
 }%
}%
```

\@classx

```
\def\@classx@array{%
  \ifcase \@lastchclass
  \@acolampacol \or
```

```
  \@addamp \@acol \or
  \@acolampacol \or
  \or
  \@acol \@firstampfalse \or
  \@addamp
  \fi
}%
\def\@classx@array@new{%
 \ifcase \@lastchclass
  \@acolampacol
 \or
  \@addamp \@acol
 \or
  \@acolampacol
 \or
 \or
  \@firstampfalse\@acoll
 \or
  \@addamp
 \fi
}%
```

6.14 Repair other broken parts of LaTeX

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

```
\def\@xbitor@LaTeX #1{\@tempcntb \count#1
   \ifnum \@tempcnta =\z@
   \else
     \divide\@tempcntb\@tempcnta
     \ifodd\@tempcntb \@testtrue\fi
   \fi}%
\def\@xbitor@ltx#1{%
 \@tempcntb\count#1%
 \@ifnum{\@tempcnta=\z@}{}{%
  \divide\@tempcntb\@tempcnta
  \@ifodd\@tempcntb{\@testtrue}{}%
 }%
}%
\@ifx{\@xbitor\@xbitor@LaTeX}{%
  \class@info{Repairing broken LaTeX \string\@xbitor}%
}{%
  \class@info{Unrecognized LaTeX \string\@xbitor. Please update this document clas
}%
\let\@xbitor\@xbitor@ltx
```

6.15 Syntax

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

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

6.16 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 4th 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 compatability with `hyperref`.

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

```
\def\@starttoc#1{%
  \begingroup
    \toc@pre
    \makeatletter
    \@input{\jobname.#1}%
    \if@filesw
      \expandafter\newwrite\csname tf@#1\endcsname
      \immediate\openout \csname tf@#1\endcsname \jobname.#1\relax
    \fi
    \@nobreakfalse
    \toc@post
  \endgroup
}%
\def\toc@pre{}%
\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.

```
\def\toc@@font{}%{\footnotesize\rmfamily}%
\def\@dotsep{\z@}%{5.5pt}%
```

`\l@section` Interface for determining which TOC elements are automatically indented.

All of the `\l@`... commands simply go through the bottleproc `\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.

```
\let\tocdim@section       \leftmargini
\let\tocdim@subsection    \leftmarginii
\let\tocdim@subsubsection \leftmarginiii
\let\tocdim@paragraph     \leftmarginiv
\let\tocdim@appendix      \leftmarginv
\let\tocdim@pagenum       \leftmarginvi
```

`\toc@pre@auto` `\toc@post@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) 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 LaTeX processing, the substyle does: `\let\toc@pre\toc@pre@auto` and `\let\toc@post\toc@post@auto`.

```
\def\toc@pre@auto{%
  \toc@@font
  \@tempdima\z@
  \toc@setindent\@tempdima{section}%
  \toc@setindent\@tempdima{subsection}%
  \toc@setindent\@tempdima{subsubsection}%
  \toc@setindent\@tempdima{paragraph}%
  \toc@letdimen{appendix}%
  \toc@letdimen{pagenum}%
}%
\def\toc@post@auto{%
  \if@filesw
   \begingroup
    \toc@writedimen{section}%
    \toc@writedimen{subsection}%
    \toc@writedimen{subsubsection}%
    \toc@writedimen{paragraph}%
    \toc@writedimen{appendix}%
    \toc@writedimen{pagenum}%
   \endgroup
  \fi
}%
```

`\toc@setindent`

```
\def\toc@setindent#1#2{%
 \csname tocdim@#2\endcsname\tocdim@min\relax
 \@ifundefined{tocmax@#2}{\@namedef{tocmax@#2}{\z@}}{}%
 \advance#1\@nameuse{tocmax@#2}\relax
 \expandafter\edef\csname tocleft@#2\endcsname{\the#1}%
}%
```

`\toc@letdimen`

```
\def\toc@letdimen#1{%
 \csname tocdim@#1\endcsname\tocdim@min\relax
 \@ifundefined{tocmax@#1}{\@namedef{tocmax@#1}{\z@}}{}%
 \expandafter\let\csname tocleft@#1\expandafter\endcsname\csname tocmax@#1\endcsna
}%
```

\toc@writedimen

```
\def\toc@writedimen#1{%
 \immediate\write\@auxout{%
  \gdef\expandafter\string\csname tocmax@#1\endcsname{%
   \expandafter\the\csname tocdim@#1\endcsname
  }%
 }%
}%
```

\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\z@ must endure over the invocation of #3.

```
\def\l@@sections#1#2#3#4{%
 % #1 - superior section
 % #2 - this section
 % #3 - content, including possible \numberline
 % #4 - page number
 \begingroup
  \everypar{}%
  \set@tocdim@pagenum{#4}%
  \global\@tempdima\csname tocdim@#2\endcsname
  \leftskip\csname tocleft@#2\endcsname\relax
  \dimen@\csname tocleft@#1\endcsname\relax
  \parindent-\leftskip\advance\parindent\dimen@
  \rightskip\tocleft@pagenum plus 1fil\relax
  \skip@\parfillskip\parfillskip\z@
  \let\numberline\numberline@@sections
  \@nameuse{l@f@#2}%
  \ignorespaces#3\unskip\nobreak\hskip\skip@
  \hb@xt@\rightskip{\hfil\unhbox\@tempboxa}\hskip-\rightskip\hskip\z@skip
  \par
  \expandafter\aftergroup\csname tocdim@#2\endcsname\expandafter
 \endgroup\the\@tempdima\relax
}%
\def\set@tocdim@pagenum#1{%
 \setbox\@tempboxa\hbox{\ignorespaces#1}%
 \@ifdim{\tocdim@pagenum<\wd\z@}{\global\tocdim@pagenum\wd\z@}{}%
}%
```

\numberline@@sections The bottleproc 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!

```
\def\numberline@@sections#1{%
 \leavevmode\hb@xt@-\parindent{%
  \hfil
  \@if@empty{#1}{}{%
   \setbox\z@\hbox{#1.\kern\@dotsep}%
   \@ifdim{\@tempdima<\wd\z@}{\global\@tempdima\wd\z@}{}%
   \unhbox\z@
  }%
```

```
 }%
 \ignorespaces
}%
\def\tocdim@min{\z@}%
```

6.17 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 `\hbox`es 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 LaTeX kernel procedure `\list`, which is the implementation of all lists.

```
\def\list#1#2{%
  \ifnum \@listdepth >5\relax
    \@toodeep
  \else
    \global\advance\@listdepth\@ne
  \fi
  \rightmargin\z@
  \listparindent\z@
  \itemindent\z@
  \csname @list\romannumeral\the\@listdepth\endcsname
  \def\@itemlabel{#1}%
  \let\makelabel\@mklab
  \@nmbrlistfalse
  #2\relax
  \@trivlist
  \parskip\parsep
  \set@listindent
  \ignorespaces
}%
\def\set@listindent@parshape{%
 \parindent\listparindent
 \advance\@totalleftmargin\leftmargin
 \advance\linewidth-\rightmargin
 \advance\linewidth-\leftmargin
 \parshape\@ne\@totalleftmargin\linewidth
}%
\def\set@listindent@{%
 \parindent\listparindent
 \advance\@totalleftmargin\leftmargin
 \advance\rightskip\rightmargin
 \advance\leftskip\@totalleftmargin
}%
\let\set@listindent\set@listindent@parshape
```

6.18 End of the `ltxutil` DOCSTRIP module

Here ends the module.

```
%</ltxutil-krn>
```

Here ends the programmer's documentation.

Index

A

B

C

I

J

K

L

M

N

O

P

U

V

W

X

Y

Z