LaTeX for ISO Standards: Source code*

Peter Wilson
Catholic University of America†
Now at peter.r.wilson@boeing.com

2002/07/22

Contents

*This file has version number v2.4, last revised 2002/07/22.

†This work was originally performed as a Guest Researcher at the National Institute of Standards and Technology.

1 Introduction

This document provides the commented source for LaTeX class and package files designed for the typesetting of documents according to the rules for ISO international standards. A seperate document provides the user manual [Wil96]. This manual is typeset according to the conventions of the LaTeX DOCSTRIP utility which enables the automatic extraction of the LaTeX macro source files [GMS94].

The original version of this class was used for the production of camera ready copy for the ISO 10303 standard *Product data representation and exchange*. The initial release of ISO 10303:1994 consisted of twelve parts and over 2400 pages. The editorial board of the ISO Central Secretariat in Geneva accepted the typographic conventions embodied in those macros.

ISO (the International Organization for Standardisation) specify their document layout requirements in ISO Directives [ISO97]. Unfortunately these Directives do not completely define the document layout, leaving several aspects open

to interpretation by the document editor and re-interpretation by the ISO editorial board. At the request of the editors of ISO 10303, and no doubt others as well, ISO has clarified the intent of their Directives [ISO01]. Also, since they were published ISO has been considering how best to accept and use electronic manuscripts instead of camer ready paper copy. At the time of writing (July 2001) they will accept documents in PDF format. This has also lead to some changes in requirements.

The following specifications are a re-implementation of the class macros published in July 2000.

This manual is provided as a service for future developers and maintainers of the class and packages for ISO standards. It is assumed that any any such person is LaTeX literate and accustomed to supporting complex class and package files [GMS94].

Sections 2 through 4 describe some administrative elements and code for general use later in the specification. The macros forming the class file are defined in sections 5 through 13. These are principally revisions of the report class to meet ISO typographic requirements and many new macros to support specific structural elements of an ISO standard to provide logical markup capabilities. Section 14 describes the macros for the `askinc` package for interactive file inclusion.

2 A driver for this document

The next series of code contains the documentation driver file for LaTeX, i.e., the file that will produce the documentation you are currently reading. This will be extracted from this file by the DOCSTRIP program.

```
⟨*driver⟩
\documentclass{ltxdoc}
```

We do not want the following basic elements to appear in the index.

```
\DoNotIndex{\',\.,\@M,\@@input,\@addtoreset,\@arabic,\@badmath}
\DoNotIndex{\@centercr,\@cite}
\DoNotIndex{\@dotsep,\@empty,\@float,\@gobble,\@gobbletwo,\@ignoretrue}
\DoNotIndex{\@input,\@ixpt,\@m}
\DoNotIndex{\@minus,\@mkboth,\@ne,\@nil,\@nomath,\@plus,\@set@topoint}
\DoNotIndex{\@tempboxa,\@tempcnta,\@tempdima,\@tempdimb}
\DoNotIndex{\@tempswafalse,\@tempswatrue,\@viipt,\@viiipt,\@vipt}
\DoNotIndex{\@vpt,\@warning,\@xiipt,\@xipt,\@xivpt,\@xpt,\@xviipt}
\DoNotIndex{\@xxpt,\@xxvpt,\\,\ ,\addpenalty,\addtolength,\addvspace}
\DoNotIndex{\advance,\Alph,\alph}
\DoNotIndex{\arabic,\ast,\begin,\begingroup,\bfseries,\bgroup,\box}
\DoNotIndex{\bullet}
\DoNotIndex{\cdot,\cite,\CodelineIndex,\cr,\day,\DeclareOption}
\DoNotIndex{\def,\DisableCrossrefs,\divide,\DocInput,\documentclass}
\DoNotIndex{\DoNotIndex,\egroup,\ifdim,\else,\fi,\em,\endtrivlist}
\DoNotIndex{\EnableCrossrefs,\end,\end@dblfloat,\end@float,\endgroup}
\DoNotIndex{\endlist,\everycr,\everypar,\ExecuteOptions,\expandafter}
\DoNotIndex{\fbox}
```

```
\DoNotIndex{\filedate,\filename,\fileversion,\fontsize,\framebox,\gdef}
\DoNotIndex{\global,\halign,\hangindent,\hbox,\hfil,\hfill,\hrule}
\DoNotIndex{\hsize,\hskip,\hspace,\hss,\if@tempswa,\ifcase,\or,\fi,\fi}
\DoNotIndex{\ifhmode,\ifvmode,\ifnum,\iftrue,\ifx,\fi,\fi,\fi,\fi,\fi}
\DoNotIndex{\input}
\DoNotIndex{\jobname,\kern,\leavevmode,\let,\leftmark}
\DoNotIndex{\list,\llap,\long,\m@ne,\m@th,\mark,\markboth,\markright}
\DoNotIndex{\month,\newcommand,\newcounter,\newenvironment}
\DoNotIndex{\NeedsTeXFormat,\newdimen}
\DoNotIndex{\newlength,\newpage,\nobreak,\noindent,\null,\number}
\DoNotIndex{\numberline,\OldMakeindex,\OnlyDescription,\p@}
\DoNotIndex{\pagestyle,\par,\paragraph,\paragraphmark,\parfillskip}
\DoNotIndex{\penalty,\PrintChanges,\PrintIndex,\ProcessOptions}
\DoNotIndex{\protect,\ProvidesClass,\raggedbottom,\raggedright}
\DoNotIndex{\refstepcounter,\relax,\renewcommand,\reset@font}
\DoNotIndex{\rightmargin,\rightmark,\rightskip,\rlap,\rmfamily,\roman}
\DoNotIndex{\roman,\secdef,\selectfont,\setbox,\setcounter,\setlength}
\DoNotIndex{\settowidth,\sfcode,\skip,\sloppy,\slshape,\space}
\DoNotIndex{\symbol,\the,\trivlist,\typeout,\tw@,\undefined,\uppercase}
\DoNotIndex{\usecounter,\usefont,\usepackage,\vfil,\vfill,\viiipt}
\DoNotIndex{\viipt,\vipt,\vskip,\vspace}
\DoNotIndex{\wd,\xiipt,\year,\z@}
```

We do want an index, using linenumbers, but not update information.

```
\EnableCrossrefs
\CodelineIndex
%% \RecordChanges
```

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

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

Some commonly used abbreviations

```
\newcommand*{\Lopt}[1]{\textsf {#1}}            % typeset an option
\newcommand*{\file}[1]{\texttt {#1}}            % typeset a file
\newcommand*{\Lcount}[1]{\textsl {\small#1}}    % typeset a counter
\newcommand*{\pstyle}[1]{\textsl {#1}}          % typeset a pagestyle
\newcommand*{\Lenv}[1]{\texttt {#1}}            % typeset an environment
\newcommand*{\Lpack}[1]{\textsf {#1}}           % typeset a package
```

We want the full details printed.

```
\begin{document}
\DocInput{isoe.dtx}
\PrintIndex
%% \PrintChanges
\end{document}
```

⟨/driver⟩

3 Identification

The `iso` document class can only be used with LaTeX2e, so we make sure that an appropriate message is displayed when another TeX format is used.

```
59 ⟨iso⟩\NeedsTeXFormat{LaTeX2e}
```

Announce the name, option files and version for LaTeX2e files:

```
60 ⟨iso⟩\ProvidesClass{isov2}[2002/07/22 v2.4 LaTeX ISO document class]
61 ⟨9pt⟩\ProvidesFile{iso9.clo}[1997/11/30 v1.1 ISO class size option]
62 ⟨10pt⟩\ProvidesFile{iso10.clo}[1997/11/30 v1.1 ISO class size option]
63 ⟨11pt⟩\ProvidesFile{iso11.clo}[1997/11/30 v1.1 ISO class size option]
64 ⟨inc⟩\ProvidesPackage{askincv1}[1995/05/31 Interactive include package]
65 ⟨fwd1⟩\ProvidesFile{isofwdbp.tex}[2002/01/10 ISO Foreword boilerplate]
66 ⟨trfwd1⟩\ProvidesFile{trfwd1.tex}[2002/01/10 PAS/TS Foreword boilerplate]
```

4 Initial Code

```
67 ⟨*iso⟩
```

The class requires the `url` package, so make sure that it is loaded.

```
68 \RequirePackage{url}
```

In this part we define a few commands that are used later on.

`\@ptsize` This control sequence is used to store the second digit of the pointsize we are typesetting in. So, normally, it's value is one of 0, 1 or 2.

```
69 \newcommand{\@ptsize}{}
```

`\if@restonecol` When the document has to be printed in two columns, we sometimes have to temporarily switch to one column. This switch is used to remember to switch back.

```
70 \newif\if@restonecol
```

`\isostringsequal` `\isoemptystring` The command `\isostringsequal` is based on code in Stephan von Bechtolsheim *TEX in Practice*, vol III page 334. It enables the definition of specific commands for testing whether two strings are equal.

```
71 \def\isostringsequal #1#2{%
72   TT\fi
73   \edef\@is@str@ngsequali{#1}%
74   \edef\@is@str@ngsequalii{#2}%
75   \ifx\@is@str@ngsequali\@is@str@ngsequalii}
```

Now we define the `\isoemptystring` command for use in testing for an empty parameter.

```
76 \def\isoemptystring #1{%
77   TT\fi
78   \if\isostringsequal{#1}{}}
```

`\fillline` This command draws a horizontal line across the page.

```
79 \newcommand{\fillline}{\mbox{}\hrulefill\mbox{}}
```

\makecommand
\make@command

The \makecommand macro is like the \newcommand macro except that it always (re)defines a command. It is equivalent to the pair of commands: \providecommand{\com}...\renewcommand{\com}....
The code for \make@command} is a simplified version of the code for \renew@command in file ltdefns.dtx.

```
\newcommand{\makecommand}{\@star@or@long\make@command}
\newcommand{\make@command}[1]{%
  \let\@ifdefinable\@rc@ifdefinable
  \new@command#1}
```

\ifpdf

This can be used to check whether or not a document is being processed by LaTeX or pdfLaTeX.

```
\newif\ifpdf
\ifx\pdfoutput\undefined
  \pdffalse
\else
  \pdftrue
\fi
```

\ifisohyper

This can be used to check, after \begin{document} to check if the hyperref package has been used.

```
\newif\ifisohyper
  \isohyperfalse
\AtBeginDocument{%
  \@ifpackageloaded{hyperref}%
      {\isohypertrue}%
      {\newcommand{\hyperpage}[1]{#1}}%
}

```

5 Declaration of Options

5.1 Setting Paper Sizes

The variables \paperwidth and \paperheight should reflect the physical paper size after trimming. For desk printer output this is usually the real paper size since there is no post-processing. We assume that the document will only be printed on either ISO standard A4 paper (option a4paper) or on the most common of the US paper sizes (option letterpaper).

Option a4paper will be the default.

\if@us

A flag for the paper size option.

```
\newif\if@us\@usfalse
```

Declare the paper size options.

```
\DeclareOption{a4paper}
   {\setlength\paperheight {297mm}%   %% 11.69in
```

```
    \setlength\paperwidth  {210mm}}   %%  8.27in
\DeclareOption{letterpaper}
   {\setlength\paperheight {11in}%    %% 279mm
    \setlength\paperwidth  {8.5in}%   %% 216mm
    \@ustrue}
```

5.2 Choosing the type size

The type size options are handled by defining `\@ptsize` to contain the last digit of the size in question and branching on `\ifcase` statements. This is done for historical reasons to stay compatible with other packages that use the `\@ptsize` variable to select special actions. It makes the declarations of size options less than 10pt difficult, although one can probably use `9` assuming that a class will not define both `9pt` and `19pt` options.

Option `11pt` will be the default.

```
\renewcommand{\@ptsize}{1}
\DeclareOption{9pt}{\renewcommand{\@ptsize}{9}}
\DeclareOption{10pt}{\renewcommand{\@ptsize}{0}}
\DeclareOption{11pt}{\renewcommand{\@ptsize}{1}}
```

5.3 Two-side or one-side printing

For two-sided printing we use the switch `\if@twoside`. In addition we have to set the `\if@mparswitch` to get any margin paragraphs into the outside margin. In this class we always use two-sided printing with marginal notes on the outside.

`\if@twoside`
`\if@mparswitch`

```
\@twosidetrue  \@mparswitchtrue
```

5.4 Two column printing

Two-column and one-column printing is again realized via a switch which is defined in the kernel. The default is single column printing.

`\if@twocolumn`

```
\DeclareOption{onecolumn}{\@twocolumnfalse}
\DeclareOption{twocolumn}{\@twocolumntrue}
```

5.5 The **copyright** option

The default is not to print ISO copyright notices. This option enables copyright notice printing. As usual, we employ a flag.

`\ifc@pyrightopt`

`c@pyrightopt` stores the user's option, while `c@pyright` will be used to control printing of copyright notices and symbols in the body of the document.

```
\newif\ifc@pyright\c@pyrightfalse
\newif\ifc@pyrightopt\c@pyrightoptfalse
```

```
\DeclareOption{copyright}{\c@pyrightopttrue}
\DeclareOption{notcopyright}{\c@pyrightoptfalse}
```

5.6 Document kind options is, dis, cd, wd, techrep, otherdoc etc.

The default is to assume that an ISO standard in preparation is to be printed (effectively this is the otherdoc option). The is option declares that an International Standard (IS) is to be printed. The fdis option declares that a Final Draft International Standard (FDIS) is to be printed, and similarly the dis option declares that a Draft International Standard (DIS) is to be printed. The cd option is for Committee Draft (CD) documents and the option wd is for Working Drafts.

The techrep option declares that a Technical Report (probably type 1 or 2) is to be printed.

The otherdoc option indicates that the document is not intended to become an ISO standard (e.g., is an ISO internal report).

\ifisstandard
\iffdisstandard
\ifdisstandard
\ifcdstandard
\ifwdstandard
\iftechrep
\ifotherdoc

We use flags for remembering which option is in effect.

```
\newif\ifisstandard\isstandardfalse
\newif\iffdisstandard\fdisstandardfalse
\newif\ifdisstandard\disstandardfalse
\newif\ifcdstandard\cdstandardfalse
\newif\ifwdstandard\wdstandardfalse
\newif\iftechrep\techrepfalse
\newif\ifotherdoc\otherdocfalse
```

\iftechspec
\ifpaspec

Flags for the techspec Technical Specification and pas Publicly Available Specification options.

```
\newif\iftechspec\techspecfalse
\newif\ifpaspec\paspecfalse
```

Now declare the options (including an is option just for completeness). We need to ensure (later) that, whatever copyright option has been used, copyright notices are not printed for certain kinds of documents.

```
\DeclareOption{is}{\isstandardtrue
                   \fdisstandardfalse
                   \disstandardfalse
                   \cdstandardfalse
                   \wdstandardfalse
                   \techrepfalse
                   \techspecfalse
                   \paspecfalse
                   \otherdocfalse}
\DeclareOption{fdis}{\isstandardfalse
                   \fdisstandardtrue
                   \disstandardfalse
                   \cdstandardfalse
                   \wdstandardfalse
```

```
                    \techrepfalse
                    \techspecfalse
                    \paspecfalse
                    \otherdocfalse}
\DeclareOption{dis}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardtrue
                    \cdstandardfalse
                    \wdstandardfalse
                    \techrepfalse
                    \techspecfalse
                    \paspecfalse
                    \otherdocfalse}
\DeclareOption{cd}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardfalse
                    \cdstandardtrue
                    \wdstandardfalse
                    \techrepfalse
                    \techspecfalse
                    \paspecfalse
                    \otherdocfalse
                    \c@pyrightfalse}
\DeclareOption{wd}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardfalse
                    \cdstandardfalse
                    \wdstandardtrue
                    \techrepfalse
                    \techspecfalse
                    \paspecfalse
                    \otherdocfalse
                    \c@pyrightfalse}
\DeclareOption{techrep}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardfalse
                    \cdstandardfalse
                    \wdstandardfalse
                    \techreptrue
                    \techspecfalse
                    \paspecfalse
                    \otherdocfalse}
\DeclareOption{techspec}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardfalse
                    \cdstandardfalse
                    \wdstandardfalse
                    \techrepfalse
                    \techspectrue
                    \paspecfalse
```

```
                    \otherdocfalse}
\DeclareOption{pas}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardfalse
                    \cdstandardfalse
                    \wdstandardfalse
                    \techreptrue
                    \techspecfalse
                    \paspectrue
                    \otherdocfalse}
\DeclareOption{otherdoc}{\isstandardfalse
                    \fdisstandardfalse
                    \disstandardfalse
                    \cdstandardfalse
                    \wdstandardfalse
                    \techrepfalse
                    \techspecfalse
                    \paspecfalse
                    \otherdoctrue
                    \c@pyrightfalse}
```

5.7 The draft option

If the user requests draft we show any overfull boxes, marginal notes are allowed, and any copyright notices are not printed. For symmetry, we also define a final option which is the default.

\ifdr@ftd@c

```
\newif\ifdr@ftd@c\dr@ftd@cfalse
\setlength{\overfullrule}{\z@}
\DeclareOption{final}{\setlength{\overfullrule}{\z@}
                      \dr@ftd@cfalse}
\DeclareOption{draft}{\setlength\overfullrule{5pt}%
                      \dr@ftd@ctrue}
```

6 Executing Options

Here we execute the default options to initialize certain variables. Note that the document class `isoe` always uses two sided printing.

```
\ExecuteOptions{notcopyright,otherdoc,final,a4paper,11pt,onecolumn}
```

The `\ProcessOptions` command causes the execution of the code for every option FOO which is declared and for which the user typed the FOO option in his `\documentclass` command. For every option BAR he typed, which is not declared, the option is assumed to be a global option. All options will be passed as document options to any `\usepackage` command in the document preamble.

```
\ProcessOptions
```

\ifc@pyright Ensure that we have the correct value of \ifc@pyright no matter the ordering in which the options are processed.

```
\c@pyrightfalse
\ifc@pyrightopt
  \c@pyrighttrue
\fi
```

Now that all the options have been executed we can load the chosen class option file that contains all size dependent code.

```
\ifnum\@ptsize < \tw@
  \input{iso1\@ptsize.clo}
\else
  \input{iso\@ptsize.clo}
\fi
```

7 Loading Packages

This class file does not load additional package files.

8 Document Layout

In this section we deal with the more difficult typographical details.

8.1 Fonts

LaTeX offers the user commands to change the size of the font, relative to the 'main' size. Each relative size changing command \size executes the command \@setfontsize\size⟨*font-size*⟩⟨*baselineskip*⟩ where:

⟨***font-size***⟩ The absolute size of the font to use from now on.

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

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

\@vpt	5	\@vipt	6	\@viipt	7
\@viiipt	8	\@ixpt	9	\@xpt	10
\@xipt	10.95	\@xiipt	12	\@xivpt	14.4
\@xviipt	17.28	\@xxpt	20.74	\@xxvpt	24.88

\normalsize \@normalsize The user level command for the main size is \normalsize. Internally LaTeX uses \@normalsize when it refers to the main size. \@normalsize will be defined to work like \normalsize if the latter is redefined from its default definition (that just issues an error message). Otherwise \@normalsize simply selects a 9pt/11pt size.

The \normalsize macro also sets new values for \abovedisplayskip, \abovedisplayshortskip and \belowdisplayshortskip.

```
⟨/iso⟩
⟨*9pt | 10pt | 11pt⟩
\renewcommand{\normalsize}{%
⟨*9pt⟩
   \@setfontsize\normalsize\@ixpt\@xpt
   \abovedisplayskip 9\p@ \@plus 2\p@ \@minus 4.5\p@
   \abovedisplayshortskip \z@ \@plus 3\p@
   \belowdisplayshortskip 5.5\p@ \@plus 2.5\p@ \@minus 3\p@
⟨/9pt⟩
⟨*10pt⟩
   \@setfontsize\normalsize\@xpt\@xiipt
   \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
⟨/10pt⟩
⟨*11pt⟩
   \@setfontsize\normalsize\@xipt{13.6}%
   \abovedisplayskip 11\p@ \@plus3\p@ \@minus6\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6.5\p@ \@plus3.5\p@ \@minus3\p@
⟨/11pt⟩
```

The \belowdisplayskip is always equal to the \abovedisplayskip. The parameters of the first level list are always given by \@listI.

```
   \belowdisplayskip \abovedisplayskip
   \let\@listi\@listI}
```

We initially choose the normalsize font.

```
\normalsize
```

\@smidgeon
\parskip
\@onelineskip

ISO typesetting is grid based, which is not something that LaTeX is good at. We use some 'fixed' skips for before and after headings, plus a flexible smidgeon.

For the grid, we want a fixed size \parskip, dependant only on the normal font, of one blank line (i.e., the \baselineskip).

Just in case the value of \parskip gets changed, also keep a similar value in @onelineskip.

```
\newlength{\@smidgeon}
  \setlength{\@smidgeon}{0.5\p@ \@plus 1\p@ \@minus 1\p@}
\newlength{\@onelineskip}
⟨9pt⟩\parskip \@xpt\p@
⟨9pt⟩\setlength{\@onelineskip}{\@xpt\p@}
⟨10pt⟩\parskip \@xiipt\p@
⟨10pt⟩\setlength{\@onelineskip}{\@xiipt\p@}
⟨11pt⟩\parskip 13.6\p@
⟨11pt⟩\setlength{\@onelineskip}{13.6\p@}
```

\small

This code is similar to that for \normalsize.

```
\newcommand{\small}{%
⟨*9pt⟩
    \@setfontsize\small\@viiipt{9}
    \abovedisplayskip 6\p@ \@plus 2\p@ \@minus 4\p@
    \abovedisplayshortskip \z@ \@plus 2\p@
    \belowdisplayshortskip 4\p@ \@plus 2\p@ \@minus 2\p@
    \def\@listi{\leftmargin\leftmargini
                \topsep 2\p@ \@plus 2\p@ \@minus 2\p@
                \parsep 1\p@ \@plus\p@ \@minus\p@
                \itemsep \parsep
                \itemindent\z@
                }%
⟨/9pt⟩
⟨*10pt⟩
   \@setfontsize\small\@ixpt{11}%
   \abovedisplayskip 8.5\p@ \@plus3\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus2\p@
   \belowdisplayshortskip 4\p@ \@plus2\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 4\p@ \@plus2\p@ \@minus2\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep
               \itemindent\z@
              }%
⟨/10pt⟩
⟨*11pt⟩
   \@setfontsize\small\@xpt\@xiipt
   \abovedisplayskip 10\p@ \@plus2\p@ \@minus5\p@
   \abovedisplayshortskip \z@ \@plus3\p@
   \belowdisplayshortskip 6\p@ \@plus3\p@ \@minus3\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 6\p@ \@plus2\p@ \@minus2\p@
               \parsep 3\p@ \@plus2\p@ \@minus\p@
               \itemsep \parsep
               \itemindent\z@
              }%
⟨/11pt⟩
   \belowdisplayskip \abovedisplayskip
}
```

`\footnotesize` This code is similar to that for `\normalsize`.

```
\newcommand{\footnotesize}{%
⟨*9pt⟩
    \@setfontsize\footnotesize\@viiipt{9}
    \abovedisplayskip 6\p@ \@plus 2\p@ \@minus 4\p@
    \abovedisplayshortskip \z@ \@plus 2\p@
    \belowdisplayshortskip 4\p@ \@plus 2\p@ \@minus 2\p@
    \def\@listi{\leftmargin\leftmargini
                \topsep 2\p@ \@plus 2\p@ \@minus 2\p@
                \parsep 1\p@ \@plus\p@ \@minus\p@
```

```
                \itemsep \parsep
                \itemindent\z@
               }%
⟨/9pt⟩
⟨*10pt⟩
   \@setfontsize\footnotesize\@viiipt{9.5}%
   \abovedisplayskip 6\p@ \@plus2\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus\p@
   \belowdisplayshortskip 3\p@ \@plus\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 3\p@ \@plus\p@ \@minus\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep
               \itemindent\z@
              }%
⟨/10pt⟩
⟨*11pt⟩
   \@setfontsize\footnotesize\@ixpt{11}%
   \abovedisplayskip 8\p@ \@plus2\p@ \@minus4\p@
   \abovedisplayshortskip \z@ \@plus\p@
   \belowdisplayshortskip 4\p@ \@plus2\p@ \@minus2\p@
   \def\@listi{\leftmargin\leftmargini
               \topsep 4\p@ \@plus2\p@ \@minus2\p@
               \parsep 2\p@ \@plus\p@ \@minus\p@
               \itemsep \parsep
               \itemindent\z@
              }%
⟨/11pt⟩
   \belowdisplayskip \abovedisplayskip
}
```

\scriptsize
\tiny
\large
\Large
\LARGE
\huge
\Huge

These are all much simpler than the previous macros, they just select a new fontsize, but leave the parameters for displays and lists alone.

```
⟨*9pt⟩
\newcommand{\tiny}{\@setfontsize\tiny\@vpt{6}}
\newcommand{\scriptsize}{\@setfontsize\scriptsize\@viipt{8}}
\newcommand{\large}{\@setfontsize\large\@xpt{11}}
\newcommand{\Large}{\@setfontsize\Large\@xipt{12}}
\newcommand{\LARGE}{\@setfontsize\LARGE\@xivpt{18}}
\newcommand{\huge}{\@setfontsize\huge\@xviipt{22}}
\newcommand{\Huge}{\@setfontsize\Huge\@xxpt{25}}
⟨/9pt⟩
⟨*10pt⟩
\newcommand{\tiny}{\@setfontsize\tiny\@vipt{7}}
\newcommand{\scriptsize}{\@setfontsize\scriptsize\@viiipt{9}}
\newcommand{\large}{\@setfontsize\large\@xipt{12}}
\newcommand{\Large}{\@setfontsize\Large\@xiipt{14}}
\newcommand{\LARGE}{\@setfontsize\LARGE\@xivpt{18}}
\newcommand{\huge}{\@setfontsize\huge\@xviipt{22}}
\newcommand{\Huge}{\@setfontsize\Huge\@xxvpt{30}}
```

```
⟨/10pt⟩
⟨*11pt⟩
\newcommand{\tiny}{\@setfontsize\tiny\@vipt{7}}
\newcommand{\scriptsize}{\@setfontsize\scriptsize\@viiipt{9}}
\newcommand{\large}{\@setfontsize\large\@xiipt{14}}
\newcommand{\Large}{\@setfontsize\Large\@xivpt{18}}
\newcommand{\LARGE}{\@setfontsize\LARGE\@xviipt{22}}
\newcommand{\huge}{\@setfontsize\huge\@xxpt{25}}
\newcommand{\Huge}{\@setfontsize\Huge\@xxvpt{30}}
⟨/11pt⟩
```

\Gfont
\Tfont
\Cfont
\SCfont
\SSCfont
\Nfont
\Efont

Define the font sizes for headings, captions, etc. `\Gfont` is the normal size font for general text. `\Tfont` is for the title of the standard. `\Cfont` is for clause headings. Similarly `\SCfont` and `\SSCfont` are for subheadings. `\Nfont` is for notes, examples, footers, footnotes, copyright. `\Efont` is for code in typewriter font.

```
\newcommand{\Gfont}{\normalsize}
\newcommand{\Nfont}{\small}
\newcommand{\Efont}{\small}
⟨*9pt⟩
%%%\newcommand{\Tfont}{\huge}
\newcommand{\Tfont}{\@setfontsize\Tfont\@xviipt{22}\bfseries}
\newcommand{\Cfont}{\Large\bfseries}
\newcommand{\SCfont}{\large\bfseries}
\newcommand{\SSCfont}{\normalsize\bfseries}

⟨/9pt⟩
⟨*10pt⟩
%%%\newcommand{\Tfont}{\huge}
\newcommand{\Tfont}{\@setfontsize\Tfont\@xviipt{22}\bfseries}
\newcommand{\Cfont}{\Large\bfseries}
\newcommand{\SCfont}{\large\bfseries}
\newcommand{\SSCfont}{\normalsize\bfseries}

⟨/10pt⟩
⟨*11pt⟩
%%%\newcommand{\Tfont}{\LARGE}
\newcommand{\Tfont}{\LARGE\bfseries}
\newcommand{\Cfont}{\Large\bfseries}
\newcommand{\SCfont}{\large\bfseries}
\newcommand{\SSCfont}{\normalsize\bfseries}

⟨/11pt⟩

```

\beforecskip
\aftercskip
\beforescskip
\afterscskip
\beforesscskip
\aftersscskip

We define skips for before and after headings. ISO wants two blank lines before a clause and one afterwards. For lower level sectioning the spacing is one blank line before and one after.

Remember that LaTeX automatically adds `\parskip` before and after headings.

```
\newlength{\beforecskip}
  \setlength{\beforecskip}{\@smidgeon}
  \addtolength{\beforecskip}{2\@onelineskip}
  \addtolength{\beforecskip}{-\parskip}
\newlength{\aftercskip}
  \setlength{\aftercskip}{\@smidgeon}
  \addtolength{\aftercskip}{\@onelineskip}
  \addtolength{\aftercskip}{-\parskip}
\newlength{\beforescskip}
  \setlength{\beforescskip}{\@smidgeon}
  \addtolength{\beforescskip}{\@onelineskip}
  \addtolength{\beforescskip}{-\parskip}
\newlength{\afterscskip}
  \setlength{\afterscskip}{\@smidgeon}
  \addtolength{\afterscskip}{\@onelineskip}
  \addtolength{\afterscskip}{-\parskip}
\newlength{\beforesscskip}
  \setlength{\beforesscskip}{\@smidgeon}
  \addtolength{\beforesscskip}{\@onelineskip}
  \addtolength{\beforesscskip}{-\parskip}
\newlength{\aftersscskip}
  \setlength{\aftersscskip}{\@smidgeon}
  \addtolength{\aftersscskip}{\@onelineskip}
  \addtolength{\aftersscskip}{-\parskip}

⟨/9pt | 10pt | 11pt⟩
⟨*iso⟩
```

`\captionsize` This internal command holds the font size for captions. Its value depends on the uglycaption option.

```
  \newcommand{\captionsize}{\normalsize}
```

8.2 Paragraphing

`\lineskip` `\normallineskip` These parameters control TEX's behaviour when two lines tend to come too close together.

```
\setlength\lineskip{1\p@}
\setlength\normallineskip{1\p@}
```

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

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

`\parindent` `\parskip` gives extra vertical space between paragraphs and `\parindent` is the width of the paragraph indentation. (`\parskip` is defined in the `.clo` file.)

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

\@lowpenalty
\@medpenalty
\@highpenalty

The commands \nopagebreak and \nolinebreak put in penalties to discourage these breaks at the point they are put in. They use \@lowpenalty, \@medpenalty or \@highpenalty, dependent on their argument.

```
\@lowpenalty   51
\@medpenalty  151
\@highpenalty 301
```

\clubpenalty
\widowpenalty

These penalties are used to discourage club and widow lines. The default values are 150 each, but we want stronger discouragement.

```
\clubpenalty  1000
\widowpenalty 1000
```

\displaywidowpenalty
\predisplaypenalty
\postdisplaypenalty

Discourage, but do not prevent, widows in front of a math display and forbid breaking directly in front of a display. Allow break after a display without a penalty. The default values are used, therefore we only show them here.

```
% \displaywidowpenalty 50
% \predisplaypenalty   10000
% \postdisplaypenalty  0
```

\interlinepenalty

Allow the breaking of a page in the middle of a paragraph.

```
% \interlinepenalty 0
```

\brokenpenalty

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

```
% \brokenpenalty 100
```

8.3 Page Layout

All margin dimensions are measured from a point one inch from the top and lefthand side of the page.

The ISO layout on A4 paper (297 by 210 mm) is 25mm sidemargins (make that 25.4mm for simplicity) 12mm above and below the header and footer, at least one blank line between the typeblock and headers/footers. This leads to \...sidemargin = 0, and \textwidth = 159.2mm = 160mm for convenience, and \topmargin = -13.5mm.

Make \headheight, \headskip and footheight each be 12pt, then \footskip = 24pt. The total height of the typeblock is then 256mm; subtracting the \topskip (say 12pt = 4mm) gives \textheight = 252mm.

8.3.1 Vertical spacing

\headheight
\headsep
\topskip

The \headheight is the height of the box that will contain the running head. The \headsep is the distance between the bottom of the running head and the top of the text. The \topskip is the \baselineskip for the first line on a page; LaTeX's output routine will not work properly if it has the value 0pt, so do not do that!

```
\setlength\headheight{12\p@}
\setlength\headsep{12\p@}
⟨/iso⟩
```

```
⟨9pt⟩\setlength\topskip{12\p@}
⟨10pt⟩\setlength\topskip{12\p@}
⟨11pt⟩\setlength\topskip{12\p@}
⟨*iso⟩
```

\footskip The distance from the baseline of the box which contains the running footer to the baseline of last line of text is controlled by the \footskip.

```
\setlength\footskip{24\p@}
```

\maxdepth \@maxdepth The TEX primitive register \maxdepth has a function that is similar to that of \topskip. The register \@maxdepth should always contain a copy of \maxdepth. In both plain TEX and LATEX 2.09 \maxdepth had a fixed value of 4pt; in native LATEX2e mode we let the value depend on the typesize. We set it so that \maxdepth + \topskip = typesize ×1.5. As it happens, in these classes \topskip is equal to the typesize, therefor we set \maxdepth to half the value of \topskip.

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

8.3.2 The dimension of text

\textwidth \textheight \columnsep The width and height of the text which are fixed in this class. Also, the gutter width when in two column mode.

```
\setlength\textwidth{160mm}
%%%\setlength\textheight{221.5mm}
\setlength\textheight{252mm}
\setlength\columnsep{10mm}
```

8.3.3 Margins

\topmargin \oddsidemargin \evensidemargin \marginparwidth \marginparsep \marginparpush The margins are fixed in this class.

```
%%%\setlength\topmargin{0mm}
\setlength\topmargin{-13.5mm}
\setlength\oddsidemargin{0mm}
\setlength\evensidemargin{0mm}
\setlength\marginparwidth{0pt}
\setlength\marginparsep{0pt}
\setlength\marginparpush{3mm}
```

However, some of the options can change these values. The draft option allows marginal notes.

```
\ifdr@ftd@c
    \setlength\marginparwidth{20mm}
    \setlength\marginparsep{0.5mm}
\fi
```

The letterpaper (279 by 216 mm) option rearranges the text block on the page, trying to center it horizontally.

```
\if@us
%%%     \setlength\topmargin{-9.4mm}
```

```
%%%    \setlength\oddsidemargin{1.55mm}
%%%    \setlength\evensidemargin{1.55mm}
    \addtolength{\topmargin}{-9mm}
    \setlength\oddsidemargin{2mm}
    \setlength\evensidemargin{2mm}
    \typeout{ }
    \typeout{*********************************************************}
    \typeout{* Warning: You have used the letterpage option. *********}
    \typeout{* This will not be acceptable as ISO camera ready copy. *}
    \typeout{*********************************************************}
    \typeout{ }
\fi
```

8.3.4 Footnotes

`\footnotesep` `\footnotesep` is the height of the strut placed at the beginning of every footnote.

```
\setlength\footnotesep{12\p@}
```

`\footins` `\skip\footins` is the space between the last line of the main text and the top of the first footnote.

```
\setlength{\skip\footins}{6\p@ \@plus 2\p@ \@minus 2\p@}
```

8.3.5 Float placement parameters

All float parameters are given default values in the LaTeX2e kernel. For this reason counters only need to be set with `\setcounter` and other parameters are set using `\renewcommand`.

Limits for the placement of floating objects

`\c@topnumber` The *topnumber* counter holds the maximum number of floats that can appear on the top of a text page (classically 2)

```
\setcounter{topnumber}{2}
```

`\topfraction` This indicates the maximum part of a text page that can be occupied by floats at the top (classically 0.7).

```
\renewcommand{\topfraction}{.8}
```

`\c@bottomnumber` The *bottomnumber* counter holds the maximum number of floats that can appear on the bottom of a text page (classically 1).

```
\setcounter{bottomnumber}{2}
```

`\bottomfraction` This indicates the maximum part of a text page that can be occupied by floats at the bottom (classically 0.3).

```
\renewcommand{\bottomfraction}{.5}
```

`\c@totalnumber` This indicates the maximum number of floats that can appear on any text page (classically 3).

```
\setcounter{totalnumber}{4}
```

`\textfraction` This indicates the minimum part of a text page that has to be occupied by text (classically 0.2).

```
\renewcommand{\textfraction}{.1}
```

`\floatpagefraction` This indicates the minimum part of a page that has to be occupied by floating objects before a 'float page' is produced (classically 0.5).

```
\renewcommand{\floatpagefraction}{.7}
```

`\c@dbltopnumber` The *dbltopnumber* counter holds the maximum number of two column floats that can appear on the top of a two column text page (classically 2).

```
\setcounter{dbltopnumber}{2}
```

`\dbltopfraction` This indicates the maximum part of a two column text page that can be occupied by two column floats at the top (classically 0.7).

```
\renewcommand{\dbltopfraction}{.8}
```

`\dblfloatpagefraction` This indicates the minimum part of a page that has to be occupied by two column wide floating objects before a 'float page' is produced (classically 0.5).

```
\renewcommand{\dblfloatpagefraction}{.7}
```

Floats on a text page

`\floatsep` `\textfloatsep` `\intextsep` When a floating object is placed on a page with text, these parameters control the separation between the float and the other objects on the page. These parameters are used for both one-column mode and single-column floats in two-column mode.

`\floatsep` is the space between adjacent floats that are moved to the top or bottom of the text page.

`\textfloatsep` is the space between the main text and floats at the top or bottom of the page.

`\intextsep` is the space between in-text floats and the text.

```
\setlength\floatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\textfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
\setlength\intextsep   {12\p@ \@plus 2\p@ \@minus 2\p@}
```

`\dblfloatsep` `\dbltextfloatsep` When floating objects that span the whole `\textwidth` are placed on a text page and LaTeX is in twocolumn mode the separation between the float and the text is controlled by `\dblfloatsep` and `\dbltextfloatsep`.

`\dblfloatsep` is the space between adjacent floats that are moved to the top or bottom of the text page.

`\dbltextfloatsep` is the space between the main text and floats at the top or bottom of the page.

```
\setlength\dblfloatsep    {12\p@ \@plus 2\p@ \@minus 2\p@}
\setlength\dbltextfloatsep{20\p@ \@plus 2\p@ \@minus 4\p@}
```

Floats on their own page or column

\@fptop
\@fpsep
\@fpbot

When floating objects are placed on separate pages the layout of such pages is controlled by these parameters. At the top of the page `\@fptop` amount of stretchable whitespace is inserted, at the bottom of the page we get an `\@fpbot` amount of stretchable whitespace. Between adjacent floats the `\@fpsep` is inserted.

These parameters are used for the placement of floating objects in one column mode, or in single column floats in two column mode.

Note that at least one of the two parameters `\@fptop` and `\@fpbot` should contain a `plus ...fil` to allow filling the remaining empty space.

```
\setlength\@fptop{0\p@ \@plus 1fil}
\setlength\@fpsep{8\p@ \@plus 2fil}
\setlength\@fpbot{0\p@ \@plus 1fil}
```

\@dblfptop
\@dblfpsep
\@dblfpbot

Double column floats in two column mode are handled with similar parameters.

```
\setlength\@dblfptop{0\p@ \@plus 1fil}
\setlength\@dblfpsep{8\p@ \@plus 2fil}
\setlength\@dblfpbot{0\p@ \@plus 1fil}
```

8.4 Page Styles

The page style *foo* is defined by defining the command `\ps@foo`. This command should make only local definitions. There should be no stray spaces in the definition, since they could lead to mysterious extra spaces in the output.

\@evenhead
\@oddhead
\@evenfoot
\@oddfoot

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

8.4.1 Marking conventions

To make headings determined by the sectioning commands, the page style defines the commands `\chaptermark`, `\sectionmark`, ...,
where `\chaptermark{`⟨*TEXT*⟩`}` is called by `\chapter` to set a mark, and so on.

The `\...mark` commands and the `\...head` macros are defined with the help of the following macros. (All the `\...mark` commands should be initialized to no-ops.)

LaTeX extends TeX's `\mark` facility by producing two kinds of marks, a 'left' and a 'right' mark, using the following commands:

`\markboth{`⟨*LEFT*⟩`}{`⟨*RIGHT*⟩`}`: Adds both marks.
`\markright{`⟨*RIGHT*⟩`}`: Adds a 'right' mark.
`\leftmark`: Used in the `\@oddhead`, `\@oddfoot`, `\@evenhead` or `\@evenfoot` macros, it gets the current 'left' mark. `\leftmark` works like TeX's `\botmark` command.

`\rightmark`: Used in the `\@oddhead`, `\@oddfoot`, `\@evenhead` or `\@evenfoot` macros, it gets the current 'right' mark. `\rightmark` works like TEX's `\firstmark` command.

The marking commands work reasonably well for right marks 'numbered within' left marks–e.g., the left mark is changed by a `\chapter` command and the right mark is changed by a `\section` command. However, it does produce somewhat anomalous results if two `\markboth`'s occur on the same page.

Commands like `\tableofcontents` that should set the marks in some page styles use a `\@mkboth` command, which is `\let` by the pagestyle command (`\ps@...`) to `\markboth` for setting the heading or to `\@gobbletwo` to do nothing.

```
%%%\mark{{}{}}   % Initializes TeX's marks   <--- can vanish
```

`\standard` `\yearofedition` `\languageofedition`

These commands are to be used in the document preamble. They are used to specify the identification of the standard, the year of the standard and the language of the standard. For example, for a DIS printed in 1995 in English:

```
\standard{ISO/DIS 10303-321}
\yearofedition{1995}
\languageofedition{(E)}
```

`\thestandard` `\thesyear` `\theslanguage` `\@runninghead`

`\thestandard` and `\thesyear` hold the number and year of the standard being documented. `\theslanguage` holds the ISO identification of the publication language; note that this must include parentheses around the code letter.

```
\gdef\thestandard{}
\gdef\thesyear{}
\gdef\theslanguage{}
\def\standard#1{\gdef\thestandard{#1}}
\def\yearofedition#1{\gdef\thesyear{#1}}
\def\languageofedition#1{\gdef\theslanguage{#1}}
```

`\@runninghead` contains the document identification text for the running head. Its value depends on the otherdoc option.

```
\ifotherdoc
```

This is not intended to be a standard, so just use `\thestandard` text.

```
    \newcommand{\@runninghead}{\thestandard}
\else
```

It either is, or is intended to become, a standard, 'so the year and language are required as well; note the colon.

```
    \newcommand{\@runninghead}{\thestandard:\thesyear\theslanguage}
\fi

```

`\copyrighthead`

`\copyrighthead` contains the text for a copyright mark in a heading. However, it should be blank if the document is not copyrighted.

```
\newcommand{\copyrighthead}{\ifc@pyright
  {\mbox{\copyright \textsc{\copyrightname} \thesyear{} --- All rights reserved}}
```

```
  \else
    \mbox{}
  \fi}

```

\extrahead `\extrahead` puts its contents into the page header (e.g., a document number). Use it in the preamble as `\renewcommand{\extrahead}{N5496}`.

```
\newcommand{\extrahead}{\mbox{}}

```

8.4.2 Defining the page styles

The pagestyles *empty* and *plain* are defined in `latex.dtx`.

\ps@headings *headings* is the typical pagestyle throughout the document. The header contains the identification of the standard. The footer has the page number at the outer edge and a copyright notice at the inner.

```
\newcommand{\ps@headings}{%
    \def\@oddhead{\bfseries\extrahead\hfil\@runninghead}%
    \def\@evenhead{\bfseries\@runninghead\hfil\extrahead}%
    \def\@oddfoot{\copyrighthead\hfil\thepage}%
    \def\@evenfoot{\thepage\hfil\copyrighthead}}
```

\ps@startpage The *startpage* page style is similar to *headings* but without a copyright notice.

```
\newcommand{\ps@startpage}{%
    \def\@oddhead{\bfseries\extrahead\hfil\@runninghead}%
    \def\@evenhead{\bfseries\@runninghead\hfil\extrahead}%
    \def\@oddfoot{\hfil\thepage}%
    \def\@evenfoot{\thepage\hfil}}
```

\ps@nohead Pagestyle *nohead* has no headers or footers.

```
\newcommand{\ps@nohead}{%
    \def\@oddhead{}%
    \def\@evenhead{}%
    \def\@oddfoot{}%
    \def\@evenfoot{}}
```

\rectoisotitlehead
\versoisotitlehead
\ps@isotitlehead *isotitlehead* is a special pagestyle for the title page of a standard. `\rectoisotitlehead` and `\versoisotitlehead` contain the relevent texts.

```
\newcommand{\rectoisotitlehead}{%
    \fillline\vspace{0.1\baselineskip}\linebreak%
    {\bfseries \uppercase{\ISname}}
%%     \mbox{\ifc@pyright\space\copyright {\scshape \copyrightname}\else
%%           \space{\scshape (\copyrightname)}\fi}
     \hfil {\bfseries \@runninghead}%
    \vspace{-0.5\baselineskip}\linebreak\fillline}
```

```
\newcommand{\versoisotitlehead}{%
    \fillline\vspace{0.1\baselineskip}\linebreak%
    {\bfseries \@runninghead} \hfil
       {\bfseries \uppercase{\ISname}}
%%     \mbox{\ifc@pyright\space\copyright {\scshape \copyrightname}\else
%%           \space{\scshape (\copyrightname)}\fi}
    \vspace{-0.5\baselineskip}\linebreak\fillline}

\def\ps@isotitlehead{%
    \def\@oddhead{\parbox{\textwidth}{\protect\rectoisotitlehead}}%
    \def\@evenhead{\parbox{\textwidth}{\protect\versoisotitlehead}}%
%%    \def\@oddfoot{\hfil\thepage}%
%%    \def\@evenfoot{\thepage\hfil}}
    \def\@oddfoot{\copyrighthead\hfil\thepage}%
    \def\@evenfoot{\thepage\hfil\copyrighthead}}
```

9 Document Markup

9.1 The title

In this class the `\title` command is somewhat different to that in the standard classes.

`\title` `\thetitle` `\introelement` `\mainelement` `\compelement`

The command `\title{`⟨*intro*⟩`}{`⟨*main*⟩`}{`⟨*comp*⟩`}` produces a macro `\thetitle` which is used when generating the first normative clause.

First define a default `\thetitle`.

```
\gdef\thetitle{}
```

Define the elements to be used in the title.

```
\newcommand{\introelement}[1]{\if\isoemptystring{#1}\else {#1 ---\newline}\fi}
\newcommand{\mainelement}[1]{#1}
\newcommand{\compelement}[1]{\if\isoemptystring{#1}\else { --- \newline #1}\fi}
```

The `\title` command starts a new recto page with arabic numbering and initialises the counters. It also uses the `isotitlehead`.

```
\renewcommand{\title}[3]{%
    \cleardoublepage\pagenumbering{arabic}%
    \setcounter{clause}{0}%
    \ifotherdoc \else %
         \protect\thispagestyle{isotitlehead}
     \fi
     \gdef\thetitle{{\Tfont \introelement{#1} %
                             \mainelement{#2} %
                             \compelement{#3}\par}}
      \if@twocolumn
         \twocolumn[\vspace*{2\baselineskip}\vbox to 35mm{\thetitle}]
      \else
          \vspace*{2\baselineskip}\vbox to 35mm{\thetitle}
      \fi}
```

9.2 The cover

ISO will produce the cover (pages 1 and 2) for any documents they publish. It can be useful for editors to be able to provide their own, informal, cover sheet.

`cover` The `cover` environment is for typesetting an informal cover sheet. there is no restriction on what can go into it, except that if used it must be the first element in the document and the contents must not exceed a single page.

```
\newenvironment{cover}{%
  \if@twocolumn
    \@restonecoltrue\onecolumn
  \else
    \@restonecolfalse
  \fi
  \setcounter{page}{1} \pagenumbering{roman}
  \thispagestyle{empty}}{%
```

A copyright notice has to go at the foot of the second page.

```
%%  \clearpage
  \thispagestyle{startpage}
  \mbox{}
  \ifc@pyright\@copyrighttext\fi
  \newpage
  \if@restonecol\twocolumn\fi}

```

9.3 Clauses

9.3.1 Building blocks

The definitions in this part of a class file usually make use of two internal macros, `\@startsection` and `\secdef`. To understand what is going on here, we describe their syntax.

The macro `\@startsection` has 6 required arguments, optionally followed by a $*$, an optional argument and a required argument:

`\@startsection`⟨*name*⟩⟨*level*⟩⟨*indent*⟩⟨*beforeskip*⟩⟨*afterskip*⟩⟨*style*⟩ optional * `[`⟨*altheading*⟩`]`⟨*heading*⟩

It is a generic command to start a section, the arguments have the following meaning:

⟨***name***⟩ The name of the user level command, e.g., 'section'.

⟨***level***⟩ A number, denoting the depth of the section – e.g., chapter=1, section = 2, etc. A section number will be printed if and only if ⟨*level*⟩ <= the value of the *secnumdepth* counter.

⟨***indent***⟩ The indentation of the heading from the left margin

⟨***beforeskip***⟩ The absolute value of this argument gives the skip to leave above the heading. If it is negative, then the paragraph indent of the text following the heading is suppressed.

⟨***afterskip***⟩ If positive, this gives the skip to leave below the heading, else it gives the skip to leave to the right of a run-in heading.

⟨***style***⟩ Commands to set the style of the heading.

∗ When this is missing the heading is numbered and the corresponding counter is incremented.

⟨***altheading***⟩ Gives an alternative heading to use in the table of contents and in the running heads. This should be present when the ∗ form is used.

⟨***heading***⟩ The heading of the new section.

A sectioning command is normally defined to `\@startsection` and its first six arguments.

The macro `\secdef` can be used when a sectioning command is defined without using `\@startsection`. It has two arguments:

`\secdef`⟨*unstarcmds*⟩⟨*starcmds*⟩

⟨***unstarcmds***⟩ Used for the normal form of a sectioning command.

⟨***starcmds***⟩ Used for the ∗-form of a sectioning command.

You can use `\secdef` as follows:

```
\def\chapter { ... \secdef \CMDA \CMDB }
\def\CMDA    [#1]#2{ ... }  % Command to define
                            % \chapter[...]{...}
\def\CMDB    #1{ ... }      % Command to define
                            % \chapter*{...}
```

9.3.2 Overview

ISO terminology uses 'clause' instead of the typical terms for subdivisions in a document, although they do use the term 'section'. Accordingly, we have defined new terms for the document sectioning commands. We also use the shorthand 'ss' for 'subsub', and so on.

LaTeX	ISO	level
chapter	clause, annex	1
section	sclause	2
subsection	ssclause	3
subsubsection	sssclause	4
paragraph	ssssclause	5
subparagraph	sssssclause	6

We also provide 'annex' commands, which are equivalent to a 'clause' command.

9.3.3 Hyperref ToC levels

\toclevel@clause
\toclevel@sclause
\toclevel@ssclause
\toclevel@sssclause
\toclevel@ssssclause
\toclevel@sssssclause
\toclevel@annex
\toclevel@index

The hyperref package needs to know ToC entry levels.

```
\newcommand*{\toclevel@clause}{1}
\newcommand*{\toclevel@sclause}{2}
\newcommand*{\toclevel@ssclause}{3}
\newcommand*{\toclevel@sssclause}{4}
\newcommand*{\toclevel@ssssclause}{5}
\newcommand*{\toclevel@sssssclause}{6}
\newcommand*{\toclevel@annex}{1}
\newcommand*{\toclevel@index}{1}

```

9.3.4 Define Counters

\c@secnumdepth

The value of the counter *secnumdepth* gives the depth of the highest-level sectioning command that is to produce section numbers.

```
\setcounter{secnumdepth}{6}
```

The macro
\newcounter{⟨*newctr*⟩}[⟨*oldctr*⟩]
defines⟨*newctr*⟩ to be a counter, which is reset to zero when counter ⟨*oldctr*⟩ is stepped. Counter ⟨*oldctr*⟩ must already be defined.

\c@annex
\c@clause
\c@fibicl@use

These counters are used for the sectioning numbers. Clause and annex are the top level document divisions.

```
\newcounter{annex}
\newcounter{clause}
\newcounter{fibicl@use}
```

\c@sclause
\c@ssclause
\c@sssclause
\c@ssssclause
\c@sssssclause

The lower level divisions get reset by higher level divisions.

```
\newcounter{sclause}[clause]
\newcounter{ssclause}[sclause]
\newcounter{sssclause}[ssclause]
\newcounter{ssssclause}[sssclause]
\newcounter{sssssclause}[ssssclause]
```

\c@yextra

We need an extra counter for the hyperref package.

```
\newcounter{yextra}

```

For any counter *CTR*, \theCTR is a macro that defines the printed version of counter *CTR*. It is defined in terms of the following macros:

\arabic{*COUNTER*} prints the value of *COUNTER* as an arabic numeral.

\roman{*COUNTER*} prints the value of *COUNTER* as a lowercase roman numeral.

\Roman{*COUNTER*} prints the value of *COUNTER* as an uppercase roman numeral.

`\alph{`*COUNTER*`}` prints the value of *COUNTER* as a lowercase letter: 1 = a, 2 = b, etc.

`\Alph{`*COUNTER*`}` prints the value of *COUNTER* as an uppercase letter: 1 = A, 2 = B, etc.

`\theannex` `\theclause` `\thefibicl@use`

The top level division numbers.

```
\renewcommand{\theannex}{\Alph{annex}}
\renewcommand{\theclause}{\arabic{clause}}
\renewcommand{\thefibicl@use}{\arabic{fibicl@use}}
```

`\thesclause` `\thessclause` `\thesssclause` `\thessssclause` `\thesssssclause`

The lower level division number representations.

```
\renewcommand{\thesclause}{\theclause.\arabic{sclause}}
\renewcommand{\thessclause}{\thesclause.\arabic{ssclause}}
\renewcommand{\thesssclause}{\thessclause.\arabic{sssclause}}
\renewcommand{\thessssclause}{\thesssclause.\arabic{ssssclause}}
\renewcommand{\thesssssclause}{\thessssclause.\arabic{sssssclause}}

```

`\theHannex` `\theHclause` `\theHsclause` `\theHssclause` `\theHsssclause` `\theHssssclause` `\theHsssssclause`

For hyperref we have to specify a similar set of number representations.

```
\newcommand{\theHannex}{\Alph{annex}}
\newcommand{\theHclause}{\arabic{clause}}
\newcommand{\theHsclause}{\theHclause.\arabic{sclause}}
\newcommand{\theHssclause}{\theHsclause.\arabic{ssclause}}
\newcommand{\theHsssclause}{\theHssclause.\arabic{sssclause}}
\newcommand{\theHssssclause}{\theHsssclause.\arabic{ssssclause}}
\newcommand{\theHsssssclause}{\theHssssclause.\arabic{sssssclause}}

```

9.3.5 Clauses

`\zerocounters`

At the start of each document division counters like for notes and examples are zeroed.

```
\newcommand{\zerocounters}{%
  \setcounter{note}{0}\setcounter{example}{0}}
```

`\@hangfrom`

Multiline clause headings are flushleft (block paragraph style).

```
\renewcommand{\@hangfrom}[1]{#1}

```

`\clause`

The command to start a new clause.

```
\newcommand{\clause}{\zerocounters
  \addtocounter{clause}{1}
  \typeout{Clause: \theclause}
  \addtocounter{clause}{-1}
  \tocskip{\tocentryskip}
  \@startsection{clause}{1}%
    {\z@}%
    {\beforecskip}%
```

```
    {\aftercskip}%
%%    {\raggedright\Cfont\bfseries}}
    {\raggedright\Cfont}}
```

`\fibicl@use` Document divisions like the Foreword and the Bibliography are effectively unnumbered clauses, but which appear in the ToC. In order to ease support for the tex4ht package, the `\fibicl@use` command is defined, but should only be used in its starred form.

```
\newcommand{\fibicl@use}{%
  \@startsection{fibicl@use}{1}%
    {\z@}%
    {\beforecskip}%
    {\aftercskip}%
%%    {\raggedright\Cfont\bfseries}}
    {\raggedright\Cfont}}
```

9.3.6 Lower level headings

These commands all make use of `\@startsection`. They also reinitialize the note and example counters.

`\sclause` `\ssclause` `\sssclause` `\ssssclause` `\sssssclause`

```
\newcommand{\sclause}{\zerocounters
    \@startsection{sclause}{2}%
        {\z@}%
        {\beforescskip}%
        {\afterscskip}%
        {\raggedright\SCfont}}

\newcommand{\ssclause}{\zerocounters
    \@startsection{ssclause}{3}%
        {\z@}%
        {\beforesscskip}%
        {\aftersscskip}%
        {\raggedright\SSCfont}}

\newcommand{\sssclause}{\zerocounters
    \@startsection{sssclause}{4}%
        {\z@}%
        {\beforesscskip}%
        {\aftersscskip}%
        {\raggedright\SSCfont}}

\newcommand{\ssssclause}{\zerocounters
    \@startsection{ssssclause}{5}%
        {\z@}%
        {\beforesscskip}%
        {\aftersscskip}%
        {\raggedright\SSCfont}}

\newcommand{\sssssclause}{\zerocounters
    \@startsection{sssssclause}{6}%
```

```
        {\z@}%
        {\beforesscskip}%
        {\aftersscskip}%
        {\raggedright\SSCfont}}
```

Preloaded definitions.

```
\def\clausemark#1{}
\def\sclausemark#1{}
\def\ssclausemark#1{}
\def\sssclausemark#1{}
\def\ssssclausemark#1{}
\def\sssssclausemark#1{}
```

9.3.7 Annexes

`\init@nnex` As an annex command has to do quite a lot, we define the internal `\init@nnex` command as a worker. It has to:

- clear the page;
- reset the table and figure counters to zero;
- redefine the `\thefigure` and the `\thetable` to precede them with the annex letter;
- reset the `sclause` counter to zero;
- test for annexes I and O since these are not allowed by ISO.

Use as: `\@annex{`⟨*title*⟩`}{`⟨*typeset body kind*⟩`}{`⟨*typeset toc kind*⟩`}`

```
\newcommand{\init@nnex}{%
    \clearpage
```

Reset the counters and test for illegal annex numbering

```
    \setcounter{table}{0}
    \setcounter{figure}{0}
    \setcounter{sclause}{0}
    \zerocounters
    \refstepcounter{annex}
    \ifnum 9=\value{annex} \refstepcounter{annex}\fi
    \ifnum 15=\value{annex} \refstepcounter{annex}\fi
```

Reset the numbering scheme, but not just when first called.

```
%%%     \ifnum 1=\value{annex}
        \renewcommand{\clause}{%
          \ClassWarning{iso}{%
            \protect\clause\space commands are not allowed after starting Annexes}{%
            Type \space <return> to proceed and I'll ignore your \protect\clause.}}
        \renewcommand{\thesclause}{\theannex.\arabic{sclause}}
        \renewcommand{\thetable}{\theannex.\arabic{table}}
        \renewcommand{\thefigure}{\theannex.\arabic{figure}}
```

```
      \renewcommand{\theHsclause}{\theHannex.\arabic{sclause}}
      \ifisohyper
        \renewcommand{\theHtable}{\theHannex.\arabic{table}}
        \renewcommand{\theHfigure}{\theHannex.\arabic{figure}}
      \fi
%%%   \fi
```

Prevent floats appearing before the title.

```
    \global\@topnum\z@
    \@afterindentfalse
}
```

`\makepreannexhead` Command to typeset the first part of an annex title. Use as `\makepreannexhead{`⟨*type*⟩`}`.

```
\newcommand{\makepreannexhead}[1]{%
  \begin{center}
  {{\Cfont \annexname~\theannex}\\\Large #1}
  \end{center}
}
```

`\makeannexhead` Typeset the title name of an annex. Use as `\makeannexhead{`⟨*title*⟩`}`.

```
\newcommand{\makeannexhead}[1]{%
  \centerline{\Cfont #1}
  \vskip 0.5\baselineskip
}
```

`\addannextotoc` Add an annex title to the ToC. Use as `\addannextotoc{`⟨*type*⟩`}{`⟨*title*⟩`}`.

```
\newcommand{\addannextotoc}[2]{%
  \tocskip{\tocentryskip}
  \addcontentsline{toc}{annex}{\ifnum2>\c@secnumdepth \else
    \protect\numberline{\annexname~\theannex\space #1}\fi #2}%
}
```

`\@infannex` `\infannex` `\@normannex` `\normannex` `\@repannex` `\repannex` Three kinds of annexes are provided. `\infannex` is an informative annex and `\normannex` is a normative annex. Just to round things out, `\repannex` is neither of these.

All the titles are centered, together with the kind of annex.

Here are the informative annex commands.

```
\newcommand{\@infannex}[1]{%
  \makepreannexhead{(\informativename)}
  \makeannexhead{#1}
  \addannextotoc{(\informativename)}{#1}
}
\newcommand{\infannex}[1]{%
  \init@nnex
  \@infannex{#1}
  \typeout{Informative annex: #1}
}
```

Here are the normative annex commands.

```
\newcommand{\@normannex}[1]{%
```

```
  \makepreannexhead{(\normativename)}
  \makeannexhead{#1}
  \addannextotoc{(\normativename)}{#1}
}
\newcommand{\normannex}[1]{%
  \init@nnex
  \@normannex{#1}
  \typeout{Normative annex: #1}
}
```

Here are the other annex commands.

```
\newcommand{\@repannex}[1]{%
  \makepreannexhead{}
  \makeannexhead{#1}
  \addannextotoc{}{#1}
}
\newcommand{\repannex}[1]{%
  \init@nnex
  \@repannex{#1}
  \typeout{Annex: #1}
}
```

9.4 Lists

9.4.1 General List Parameters

The following commands are used to set the default values for the list environment's parameters. See the LaTeX manual for an explanation of the meanings of the parameters. Defaults for the list environment are set as follows. First, `\rightmargin`, `\listparindent` and `\itemindent` are set to 0pt. Then, for a Kth level list, the command `\@listK` is called, where 'K' denotes 'i', ''i', ... , 'vi'. (I.e., `\@listiii` is called for a third-level list.) By convention, `\@listK` should set `\leftmargin` to `\leftmarginK`.

`\leftmargin` `\leftmargini` `\leftmarginii` `\leftmarginiii` `\leftmarginiv` `\leftmarginv` `\leftmarginvi`

For efficiency, level-one list's values are defined at top level, and `\@listi` is defined to set only `\leftmargin`.

```
\setlength{\leftmargini}{2em}
```

The value of `\leftmargin` has to be set at this outer level.

```
\leftmargin \leftmargini
```

For ISO, all lists are indented the same amount.

```
\setlength{\leftmarginii}{\leftmargini}
\setlength{\leftmarginiii}{\leftmargini}
\setlength{\leftmarginiv}{\leftmargini}
\setlength{\leftmarginv}{\leftmargini}
\setlength{\leftmarginvi}{\leftmargini}
```

`\itemindent`

Here we set the `\itemindent` which is the extra indentation before a label.

```
\setlength{\itemindent}{\z@}
```

\labelsep
\labelwidth

\labelsep is the distance between the label and the text of an item; \labelwidth is the width of the label.

```
\setlength{\labelsep}{0.5em}
\setlength{\labelwidth}{\leftmargini}
  \addtolength{\labelwidth}{-\labelsep}
```

\partopsep

When the user leaves a blank line before the environment an extra vertical space of \partopsep is inserted, in addition to \parskip and \topsep.

```
⟨/iso⟩
⟨*9pt | 10pt | 11pt⟩
\setlength\partopsep{2\p@ \@plus 1\p@ \@minus 1\p@}
⟨/9pt | 10pt | 11pt⟩
⟨*iso⟩
```

\@beginparpenalty
\@endparpenalty

These penalties are inserted before and after a list or paragraph environment. They are set to a bonus value to encourage page breaking at these points.

\@itempenalty

This penalty is inserted between list items.

```
\@beginparpenalty -\@lowpenalty
\@endparpenalty   -\@lowpenalty
\@itempenalty     -\@lowpenalty
```

\@setitemparams

Lists may be called within other list environments with differing layouts. We use a routine to set the layout for itemize and enumerate lists.

```
⟨/iso⟩
⟨*9pt | 10pt | 11pt⟩
\newcommand{\@setitemparams}{%
  \setlength{\labelsep}{0.5em}
  \setlength{\labelwidth}{\leftmargini}
    \addtolength{\labelwidth}{-\labelsep}
  \setlength{\itemindent}{\z@}
  \setlength{\parsep}{\baselineskip}
  \topsep \z@ \@plus1\p@ \@minus1\p@
  \itemsep \z@ \@plus1\p@ \@minus1\p@}
```

\@listI
\@listi

\@listI defines top level and \@listi values of \leftmargin, \parsep, \topsep, and \itemsep

```
\def\@listi{\leftmargin\leftmargini
 %%%            \itemindent\labelsep
 %%             \itemindent\z@
 %%           \parsep\baselineskip
 %%           \topsep 0\p@ \@plus1\p@ \@minus1\p@
 %%           \itemsep0\p@ \@plus1\p@ \@minus1\p@}
  \@setitemparams}
\let\@listI\@listi
```

We have to initialise these parameters.

```
\@listi
```

\@listii \@listiii \@listiv \@listv \@listvi

Here are the same macros for the lower level lists.

```
\def\@listii {\leftmargin\leftmarginii
 %%%              \itemindent\labelsep}
 %%             \itemindent\z@
  \@setitemparams
              }
\def\@listiii{\leftmargin\leftmarginiii
 %%%              \itemindent\labelsep}
 %%             \itemindent\z@
  \@setitemparams
              }
\def\@listiv {\leftmargin\leftmarginiv
 %%%              \itemindent\labelsep}
 %%             \itemindent\z@
  \@setitemparams
              }
\def\@listv  {\leftmargin\leftmarginv
 %%%              \itemindent\labelsep}
 %%             \itemindent\z@
  \@setitemparams
              }
\def\@listvi {\leftmargin\leftmarginvi
 %%%              \itemindent\labelsep}
 %%             \itemindent\z@
  \@setitemparams
              }
⟨/9pt | 10pt | 11pt⟩
⟨*iso⟩
```

9.4.2 Enumerate

ISO only requires two levels of enumeration labelled 'a)' and '1)'. We include a third level and fourth labelled 'i)' and 'A)', just in case. ISO has printed ISO 10303:1994 which includes all three levels defined here. The enumerate environment uses four counters: *enumi*, *enumii*, *enumiii* and *enumiv*, where *enumN* controls the numbering of the Nth level enumeration.

\theenumi \theenumii \theenumiii \theenumiv

The counters are already defined in `latex.dtx`, but their representation is changed here.

```
\renewcommand{\theenumi}{\alph{enumi}}
\renewcommand{\theenumii}{\arabic{enumii}}
\renewcommand{\theenumiii}{\roman{enumiii}}
\renewcommand{\theenumiv}{\Roman{enumiv}}
```

\labelenumi \labelenumii \labelenumiii \labelenumiv

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

```
\newcommand{\labelenumi}{\theenumi)}
\newcommand{\labelenumii}{\theenumii)}
\newcommand{\labelenumiii}{\theenumiii)}
```

```
\newcommand{\labelenumiv}{\theenumiv)}
```

\p@enumii
\p@enumiii
\p@enumiv

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

```
\renewcommand{\p@enumii}{\theenumi}
\renewcommand{\p@enumiii}{\p@enumii\theenumii}
\renewcommand{\p@enumiv}{\p@enumiii\theenumiii}
```

enumerate

We modify the default enumerate environment to make labels flush left in the label box.

```
\def\enumerate{%
  \ifnum \@enumdepth >\thr@@\@toodeep\else
    \advance\@enumdepth\@ne
    \edef\@enumctr{enum\romannumeral\the\@enumdepth}%

      \expandafter
      \list
        \csname label\@enumctr\endcsname
        {\usecounter\@enumctr\def\makelabel##1{##1\hfill}}%
  \fi}
\let\endenumerate =\endlist
```

9.4.3 Itemize

ISO only requires one level labelled with either a long dash or a bullet. We provide four levels, three of which have been used in ISO 10303:1994.

\labelitemi
\labelitemii
\labelitemiii
\labelitemiii

Itemization is controlled by the commands: \labelitemi, \labelitemii, etc., which define the labels of the various itemization levels: the symbols used are bold em-dash, bullet, asterisk, and centered period.

```
\newcommand{\labelitemi}{{\normalfont\bfseries \textemdash}\hfill}
%%\newcommand{\labelitemii}{\textbullet\hfill}
%%\newcommand{\labelitemiii}{\textasteriskcentered}
%%\newcommand{\labelitemiv}{\textperiodcentered}
\newcommand{\labelitemii}{\labelitemi}
\newcommand{\labelitemiii}{\labelitemi}
\newcommand{\labelitemiv}{\labelitemi}
```

itemize

We modify the default itemize environment to make the labels flush left in the label box.

```
\def\itemize{%
  \ifnum \@itemdepth >\thr@@\@toodeep\else
    \advance\@itemdepth\@ne
    \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%

    \expandafter
    \list
      \csname\@itemitem\endcsname
```

```
       {\def\makelabel##1{##1\hfill}}%
  \fi}
\let\enditemize =\endlist
```

9.4.4 Description

`description` The description environment is defined here – while the default itemize and enumerate environments are defined in `latex.dtx`.

```
\newenvironment{description}%
               {\list{}{\labelwidth\z@ \itemindent 0.5em \labelsep 0.5em
                        \let\makelabel\descriptionlabel}}%
               {\endlist}
```

`\descriptionlabel` To change the formatting of the label, you must redefine `\descriptionlabel`. Note that the label includes a colon.

```
\newcommand*{\descriptionlabel}[1]{\normalfont\bfseries #1:\hfill}
```

9.5 Defining new environments

9.5.1 Quotation

This is not required by ISO, but we leave it in anyway.

`quotation` The quotation environment is defined by making clever use of the list environment's parameters. The lines in the environment are set smaller than `\textwidth`. The first line of a paragraph inside this environment is indented.

```
\newenvironment{quotation}%
               {\list{}{\listparindent 1.5em%
                        \itemindent    \listparindent
                        \rightmargin   \leftmargin
                        \parsep        \z@ \@plus\p@}%
                \item[]}%
               {\endlist}
```

9.5.2 Quote

This is also not an ISO requirement, but leave it in anyway.

`quote` The quote environment is like the quotation environment except that paragraphs are not indented.

```
\newenvironment{quote}%
               {\list{}{\rightmargin\leftmargin}%
                \item[]}%
               {\endlist}
```

9.5.3 Theorem

This document class does not define it's own theorem environments, the defaults, supplied by `latex.dtx` are available.

9.5.4 Notes

ISO requires that information which is essential to the understanding of a standard but which is not a requirement is to be given in the form of a note. In the Directives edition 2, there were three styles of note:

1. isolated notes which are marked NOTE - 1, NOTE - 2, etc.
2. a local grouping of notes marked
   NOTES
   1 - . . .
   2 - . . .
3. an isolated note that is not numbered because it is the only one in that (sub-) clause of the document.

The 3rd edition removed the local grouping.

`\ifinfloat` Special consideration has to be given when notes appear within a float.

```
\newif\ifinfloat\infloatfalse
```

`\c@note` `\c@floatnote` `\thenote` `\thefloatnote` Define note counters, where the counter *note* for body notes gets reset within each new clause and notes within floats have their own numbering scheme via *floatnote*.

```
\newcounter{note}[clause]
\renewcommand{\thenote}{\arabic{note}}
\newcounter{floatnote}
\renewcommand{\thefloatnote}{\arabic{floatnote}}
```

`\theHnote` `\theHfloatnote` We also need hyperref representations.

```
\newcommand{\theHnote}{\thenote.\arabic{yextra}}
\newcommand{\theHfloatnote}{\thefloatnote.\arabic{yextra}}

```

`\notelabel` Labeling of notes (and examples).

```
\newcommand{\notelabel}[1]{{#1\hfill}}
```

`notes` This environment produced a fixed heading followed by a numbered list. The environment is defined in terms of a general list.

Use as:

```
\begin{notes}
\begin{note}Text of first note ... \end{note}
\begin{note}Text of second note ... \end{note}
\end{notes}
```

With the 3rd edition of the ISO Directives, this has been made a no-op and is only retained for compatability. The original code was:

```
\newif\ifinnotes\innotesfalse
\newenvironment{notes}{\list{}%
```

```
{\ifinfloat \leftmargin 0em \else \leftmargin 2em\fi
 \itemindent 0.5em \labelwidth 0em
 \labelsep 0.5em \listparindent 0em
 \let\makelabel\notelabel}
\innotestrue
\Nfont\item[\notesname]\mbox{}\nopagebreak[2]}%
{\innotesfalse\endlist}
```

`\@setnoteparams` Because notes, and examples, have the same basic layout we use a routine to set the various parameters.

```
\newcommand{\@setnoteparams}{%
  \setlength{\partopsep}{\z@}
  \setlength{\topsep}{\z@}
  \setlength{\labelsep}{1em}
  \setlength{\itemindent}{\labelsep}
  \setlength{\labelwidth}{\z@}
  \setlength{\listparindent}{\z@}
  \setlength{\leftmargin}{\z@}     % added in v2.3
}
```

`anote` An isolated un-numbered note.

```
\newenvironment{anote}{\list{}{%
%%  \ifinfloat \setlength{\leftmargin}{\z@} \else
%%             \setlength{\leftmargin}{2em} \fi
  \@setnoteparams}
  \Nfont\item[\notename]}%
{\endlist}
```

`note` A numbered note.

```
\newenvironment{note}{\list{}{%
```

Use the appropriate counter: normally *note* but *floatnote* when in a floating environment.

```
  \stepcounter{yextra}
  \ifinfloat
    \refstepcounter{floatnote}
    \let\thenote\thefloatnote
  \else
    \refstepcounter{note}
  \fi
```

Originally we adjusted the margins according to whether we were in a notes environment or not.

```
%%  \ifinfloat \setlength{\leftmargin}{\z@} \else
%%             \setlength{\leftmargin}{2em} \fi
  \@setnoteparams}
  \Nfont\item[\notename~\thenote]}%
  {\endlist}
```

9.5.5 Examples

ISO Directives part 3 (2nd edition) had no rules on how to display an example, but it did use examples itself; these examples were displayed in a format similar to notes.

We provided two styles of example:

1. isolated examples which are marked EXAMPLE - 1, EXAMPLE - 2, etc.

2. a local grouping of examples marked
   EXAMPLES
   1 - . . .
   2 - . . .

The 3rd edition of the Directives does specify some options for typesetting examples. A single example in a (sub) clause is preceeded by the word 'EXAMPLE'. If there are several examples, then each is numbered (e.g., 'EXAMPLE 3'). It also states that all lines of an example shall be inset from the margin or set in a smaller font, so that its extent can be determined.

For now, we choose both options.

Implementation is very similar to that for notes.

`\c@example` `\theexample` `\theHexample`

Define example counter. Example numbering is only continuous within a (sub) clause (we used to have it continuous throughout the document).

```
\newcounter{example}[clause]
\renewcommand{\theexample}{\arabic{example}}
\newcommand{\theHexample}{theexample.\arabic{yextra}}
```

`examples`

Originally, this environment produces a fixed heading followed by a numbered list. The environment is defined in terms of a general list.

Use as:

```
\begin{examples}
\begin{example}Text of first ...\end{example}
\begin{example}Text of second ... \end{example}
\end{examples}
```

With the 3rd edition of the ISO Directives the environment has been made a no-op, but is retained for compatibility. The code used to be:

```
\newif\ifinexamples\inexamplesfalse
\newenvironment{examples}{\list{}%
   {\leftmargin 2em
    \itemindent 0.5em \labelwidth 0em
    \labelsep 0.5em \listparindent 0em
    \let\makelabel\notelabel}
   \inexamplestrue
   \Nfont\item[\examplesname]\mbox{}\nopagebreak[2]}%
   {\inexamplesfalse\endlist}
```

anexample An isolated un-numbered example.

```
\newenvironment{anexample}{\list{}{%
%%  \ifinfloat \setlength{\leftmargin}{\z@} \else
%%             \setlength{\leftmargin}{2em} \fi
  \@setnoteparams}
  \Nfont\item[\examplename]}{\endlist}
```

example Like the note environment.

```
\newenvironment{example}{\list{}{%
  \stepcounter{yextra}
  \refstepcounter{example}
%%  \ifinfloat \setlength{\leftmargin}{\z@} \else
%%             \setlength{\leftmargin}{2em} \fi
  \@setnoteparams}
  \Nfont\item[\examplename~\theexample]}%
  {\endlist}
```

9.5.6 Listing of references

ISO has three kinds of literature references, broken into two categories. The categories are normative and informative references. Within the normative category, references are to either published or 'unpublished' standards (IS or DIS in ISO terminology).

nreferences The nreferences environment is for listing normative references. It is implemented as a list.

\nreferencelabel Labelling of normative references.

```
\newcommand{\nreferencelabel}[1]{#1,\hfill}
```

Define the environment. It is used as:

```
\begin{nreferences}
\isref{id}{published standard title}
\disref{id}{unpublished standard title}
....
\end{nreferences}
```

```
\newenvironment{nreferences}{\list{}%
    {\leftmargin 0pt \itemindent 0.5em
     \labelwidth\z@ \labelsep 0.5em
     \let\makelabel\nreferencelabel}}%
    {\endlist}
```

\isref This is a two parameter command for printing a normative reference to a published standard.

```
\newcommand{\isref}[2]{\item[#1]{\itshape #2}}
```

\disref This is a two parameter command for printing a normative reference to an unpublished standard. ISO requires that each unpublished standard should be footnoted as 'unpublished'. Awkwardly, only one footnote is permitted. This means we have to fiddle with the footnote counter.

\ifd@is A flag to denote if there have been any previous disrefs.

```
\newif\ifd@is\d@isfalse
```

Now define the \disref command.

```
\newcommand{\disref}[2]{\begingroup
    \ifd@is
```

This is not the first call to \disref, so just footnotemark the entry

```
        {\item[#1\protect\@footnotemark]{\itshape #2}}
    \else
```

This is the first call, so we have to make the footnote

```
        \addtocounter{footnote}{1}
        \xdef\@thefnmark{\thefootnote}
        \item[#1\protect\@footnotemark]{\itshape #2}%
        \footnotetext[\value{footnote}]{\tbpname}
        \d@istrue
    \fi
    \endgroup\d@istrue}
```

references The references environment is for listing informative references. It is implemented as a list.

\c@infrefctr
\p@infrefctr
\theinfrefctr
\labelinfref

Informative references are labelled with a number enclosed in square brackets.

In the body of the text, a reference to an informatively listed document n has to be printed as [n]. Use the standard LaTeX \label command and the\bref command for this.

```
\newcounter{infrefctr}
\renewcommand{\p@infrefctr}{}
\renewcommand{\theinfrefctr}{\arabic{infrefctr}}
\newcommand{\labelinfref}{[\arabic{infrefctr}]}
```

Define the environment. It is used as:

```
\begin{references}
\reference{authors}{title}{publisher and date}
....
\end{references}
```

```
\newenvironment{references}{\list{\labelinfref}{\usecounter{infrefctr}
     \leftmargin 0pt \itemindent 0.5em
     \labelwidth\z@ \labelsep 0.5em}}%
    {\endlist}
```

`\reference` This is a three parameter command for printing an informatively listed reference document.
`\reference{`⟨*authors*⟩`}{`⟨*title*⟩`}{`⟨*publisher and date*⟩`}`

```
953 \newcommand{\reference}[3]{\item {#1} {{\itshape #2}} {#3}}
```

9.5.7 Listing of definitions

One element of an ISO standard is the listing of definitions of terms.

`olddefinitions` The `olddefinitions` environment is for listing terms which have been defined in some other standard. It is defined in terms of the `itemize` environment.

```
954 \newenvironment{olddefinitions}%
955     {\begin{itemize}}%
956     {\end{itemize}}
```

`\olddefinition` Within an `olddefinitions` environment each term is specified by the `\olddefinition{`⟨*phrase*⟩`}{`⟨*supplement*⟩`}` command.

```
957 \newcommand{\olddefinition}[2]{\item #1 #2}
```

`definitions` Terms being defined within the current document are listed within the `definitions` environment. ISO requires that each definition be sequentially numbered within the clause in which it is defined. This numbering is as though the definition formed a sub-clause.

`\c@cl@level` A counter for determing the current sectioning level.

```
958 \newcounter{cl@level}
```

`\@defcl` We use this internally for the `\definition` command. A default definition is suplied here as we are going to renew it, possibly several times.

```
959 \newcommand{\@defcl}[1]{}
```

Now we define the `definitions` environment.

```
960 \newenvironment{definitions}{%
```

First, set the *cl@level* according to the sectioning level within which the environment is called.

```
961   \setcounter{cl@level}{6}
962   \ifnum\value{ssssssclause}=0 \setcounter{cl@level}{5} \fi
963   \ifnum\value{sssssclause}=0 \setcounter{cl@level}{4} \fi
964   \ifnum\value{ssssclause}=0 \setcounter{cl@level}{3} \fi
965   \ifnum\value{sssclause}=0 \setcounter{cl@level}{2} \fi
966   \ifnum\value{ssclause}=0 \setcounter{cl@level}{1} \fi
967   \ifnum\value{sclause}=0 \setcounter{cl@level}{0} \fi
```

Now redefine an appropriate (s)clause definition to get a number on one line, followed by the heading on the next line with a bold normal font. A new paragraph is not started after the heading, and there is no entry in the ToC. As this is done within the group automatically set up be the environment, any original definitions will get restored afterwards.

```
\ifcase\value{cl@level} % 0, NOT YET IN A CLAUSE
  \ClassWarning{iso}{Definitions started before the initial clause}
  \renewcommand{\@defcl}[1]{\setcounter{note}{0}\setcounter{example}{0}
    \par
    \addvspace{\beforecskip}
    \@afterindentfalse
    \refstepcounter{clause}
    {\raggedright\bfseries \theclause\\ ##1\\}}
```

Do similar things for the other cases.

```
\or % 1, called in a clause
  \renewcommand{\@defcl}[1]{\setcounter{note}{0}\setcounter{example}{0}
    \par
    \addvspace{\beforescskip}
    \@afterindentfalse
    \refstepcounter{sclause}
    {\raggedright\bfseries \thesclause\\ ##1\\}}
\or % 2, called in an sclause
  \renewcommand{\@defcl}[1]{\setcounter{note}{0}\setcounter{example}{0}
    \par
    \addvspace{\beforesscskip}
    \@afterindentfalse
    \refstepcounter{ssclause}
    {\raggedright\bfseries \thessclause\\ ##1\\}}
\or % 3, called in an ssclause
  \renewcommand{\@defcl}[1]{\setcounter{note}{0}\setcounter{example}{0}
    \par
    \addvspace{\beforesscskip}
    \@afterindentfalse
    \refstepcounter{sssclause}
    {\raggedright\bfseries \thesssclause\\ ##1\\}}
\or % 4, called in an sssclause
  \renewcommand{\@defcl}[1]{\setcounter{note}{0}\setcounter{example}{0}
    \par
    \addvspace{\beforesscskip}
    \@afterindentfalse
    \refstepcounter{ssssclause}
    {\raggedright\bfseries \thessssclause\\ ##1\\}}
\or % 5, called in an ssssclause
  \renewcommand{\@defcl}[1]{\setcounter{note}{0}\setcounter{example}{0}
    \par
    \addvspace{\beforesscskip}
    \@afterindentfalse
    \refstepcounter{sssssclause}
    {\raggedright\bfseries \thesssssclause\\ ##1\\}}
\else % 5+, called in an sssssclause or lower
  \ClassWarning{iso}{Definitions too deeply nested}
  \renewcommand{\@defcl}[1]{
    \par
    \addvspace{\beforesscskip}
```

```
      \@afterindentfalse
      \refstepcounter{ssssssclause}
      {\raggedright\bfseries \thesssssssclause\\ ##1\\}}
  \fi}%
  {}
```

\definition Within a `definitions` environment the command `\definition{`⟨*phrase*⟩`}{`⟨*definition text*⟩`}` is used to specify and define each term. It uses the sectional heading definition stored in `\@defcl` set up by the environment.

```
\newcommand{\definition}[2]{\@defcl{#1} #2}
```

9.5.8 Listing of symbols and abbreviations

Another possible element in a standard is the listing of symbols and abbreviations. This is similar to the original `definitions` listing, except that terms are not treated as clauses.

symbols

\symbollabel

```
\newcommand{\symbollabel}[1]{{#1 \hfill}}
\newenvironment{symbols}{\list{}%
    {\itemindent 0em \leftmargin 8em
     \labelsep 1em \labelwidth 5em
     \let\makelabel\symbollabel}}%
    {\endlist}
```

\symboldef Within a `symbols` environment the command `\symboldef{`⟨*symbol*⟩`}{`⟨*meaning*⟩`}` is used to specify and explain each symbol or abbreviation.

```
\newcommand{\symboldef}[2]{\item[#1] #2}
```

9.5.9 Listing of scope items

Another possible element in a standard is the listing of items that are within the scope; conversely, listing of items that are out of scope may also be useful.

inscope
outofscope

We define synonyms for the `itemize` list environment, and initiate the lists with some boilerplate. Use as, for example:

```
\begin{inscope}{international standard}
  \item ...
  \item ...
\end{inscope}
```

```
\newenvironment{inscope}[1]{%
  \inscopename #1:
  \begin{itemize}}%
  {\end{itemize}}
\newenvironment{outofscope}[1]{%
  \outofscopename #1:
  \begin{itemize}}%
  {\end{itemize}}
```

9.6 Setting parameters for existing environments

9.6.1 Array and tabular

\arraycolsep The columns in an array environment are separated by 2\arraycolsep.

```
\setlength\arraycolsep{4\p@}
```

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

```
\setlength\tabcolsep{4\p@}
```

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

```
\setlength\arrayrulewidth{.4\p@}
```

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

```
\setlength\doublerulesep{2\p@}
```

9.6.2 Tabbing

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

```
\setlength\tabbingsep{\labelsep}
```

9.6.3 Minipage

\@minipagerestore The macro \@minipagerestore is called upon entry to a minipage environment to set up things that are to be handled differently inside a minipage environment. In the current styles, it does nothing.

\@mpfootins Minipages have their own footnotes; \skip\@mpfootins plays same rôle for footnotes in a minipage as \skip\footins does for ordinary footnotes.

```
\skip\@mpfootins = \skip\footins
```

9.6.4 Framed boxes

\fboxsep The space left by \fbox and \framebox between the box and the text in it.

\fboxrule The width of the rules in the box made by \fbox and \framebox.

```
\setlength\fboxsep{3\p@}
\setlength\fboxrule{.4\p@}
```

9.6.5 Equation and eqnarray

`equation` and `eqnarray` counters are not required by ISO, and the equations are to be left-justified. The default is for the left-hand side of equations to be flushleft.

`\theequation` The equation counter will be reset at beginning of a new chapter and the equation number will be prefixed by the chapter number.

This code must follow the `\chapter` definition, or more exactly the definition of the chapter counter.

```
1045 \renewcommand{\theequation}{\arabic{equation}}
```

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

```
1046 % \setlength\jot{3pt}
```

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

```
1047 % \def\@eqnnum{(\theequation)}
```

9.7 Floating objects

The file `latex.dtx` only defines a number of tools with which floating objects can be defined. This is done in the document class. It needs to define the following macros for each floating object of type `TYPE` (e.g., `TYPE` = figure).

`\fps@TYPE` The default placement specifier for floats of type `TYPE`.

`\ftype@TYPE` The type number for floats of type `TYPE`. Each `TYPE` has associated a unique positive `TYPE` number, which is a power of two. E.g., figures might have type number 1, tables type number 2, programs type number 4, etc.

`\ext@TYPE` The file extension indicating the file on which the contents list for float type `TYPE` is stored. For example, `\ext@figure` = 'lof'.

`\fnum@TYPE` A macro to generate the figure number for a caption. For example, `\fnum@TYPE` == 'Figure `\thefigure`'.

`\@makecaption`⟨***num***⟩⟨***text***⟩ A macro to make a caption, with ⟨*num*⟩ the value produced by `\fnum@...` and ⟨*text*⟩ the text of the caption. It can assume it's in a `\parbox` of the appropriate width. This will be used for *all* floating objects.

The actual environment that implements a floating object such as a figure is defined using the macros `\@float` and `\end@float`, which are defined in `latex.dtx`.

An environment that implements a single column floating object is started with `\@float{TYPE}[`⟨*placement*⟩`]` of type `TYPE` with ⟨*placement*⟩ as the placement specifier. The default value of ⟨*PLACEMENT*⟩ is defined by `\fps@TYPE`.

The environment is ended by `\end@float`. E.g., `\figure` == `\@float`figure, `\endfigure` == `\end@float`.

9.7.1 Figure

Here is the implementation of the figure environment.

`\c@figure` First we have to allocate a counter to number the figures. In this class figures are numbered sequentially.

```
\newcounter{figure}
\renewcommand{\thefigure}{\@arabic\c@figure}
```

`\fps@figure` `\ftype@figure` `\ext@figure` `\fnum@figure` Here are the parameters for the floating objects of type 'figure'.

```
\def\fps@figure{tbp}
\def\ftype@figure{1}
\def\ext@figure{lof}
\def\fnum@figure{\figurename~\thefigure}
```

`\iffigs` We define a flag to tell whether the document contains any figures. Elsewhere a flag, `\ifinfloat`, is defined to tell if we are in a float.

```
\newif\iffigs\figsfalse
```

`\@initisofig` At the start of a `figure` environment we have to set a flag and do some work to deal with the ISO requirements for the ToC, and also zero the floatnote counter.

```
\newcommand{\@initisofig}{%
  \iffigs\else\figstrue
    \if@filesw \immediate\write\@mainaux{%
      \string\gdef\string\setfigs{%
         \string\floatlist{\listfigurename}{lof}}}
    \fi
  \fi
```

Now deal with the possibility that the float may contain notes.

```
  \infloattrue\setcounter{floatnote}{0}
}
```

`figure` `figure*` This is the definition of the actual environment. The form with the `*` is used for double column figures.

```
\newenvironment{figure}{%
  \@initisofig
  \@float{figure}}%
```

At the end of the environment we are no longer in a float.

```
  {\end@float\infloatfalse}
```

The starred version is similar.

```
\newenvironment{figure*}{%
  \@initisofig
  \@dblfloat{figure}}%
 {\end@dblfloat\infloatfalse}
```

9.7.2 Table

Here is the implementation of the table environment. It is very much the same as the figure environment, the additional complication being that we have to flag that we are in a table, as well as being in a float.

`\c@table` First we have to allocate a counter to number the tables. In this class tables are numbered sequentially.

```
\newcounter{table}
\renewcommand{\thetable}{\@arabic\c@table}
```

`\fps@table` Here are the parameters for the floating objects of type 'table'.

`\ftype@table`
`\ext@table`
`\fnum@table`

```
\def\fps@table{tbp}
\def\ftype@table{2}
\def\ext@table{lot}
\def\fnum@table{\tablename~\thetable}
```

`\iftabs` We define a flag to tell whether the document contains any tables. Elsewhere a flag, `\ifinfloat`, is defined to tell if we are in a float.

```
\newif\iftabs\tabsfalse
```

`\@initisotab` Initial code at the start of a `table` environment.

```
\newcommand{\@initisotab}{%
  \iftabs\else\tabstrue
    \if@filesw \immediate\write\@mainaux{%
      \string\gdef\string\settabs{%
        \string\floatlist{\listtablename}{lot}}}
    \fi
  \fi
  \infloattrue\setcounter{floatnote}{0}
}
```

`table` This is the definition of the actual environment. The form with the `*` is used for `table*` double column tables.

```
\newenvironment{table}{%
  \@initisotab
  \@float{table}}%
 {\end@float\infloatfalse}
```

The starred version is similar.

```
\newenvironment{table*}{%
  \@initisotab
  \@dblfloat{table}}%
 {\end@dblfloat\infloatfalse}
```

9.7.3 A bottom float

We define an additional float environment. Unless something additional is done, this will not be listed in the table of contents.

\c@bottomfloat
\thebottomfloat
First we have to allocate a counter to number the float.

```
\newcounter{bottomfloat}
\renewcommand{\thebottomfloat}{\@arabic\c@bottomfloat}
```

\fps@bottomfloat
\ftype@bottomfloat
\ext@bottomfloat
\fnum@bottomfloat
Here are the parameters for the floating objects of type 'bottomfloat'.

```
\def\fps@bottomfloat{b}
\def\ftype@bottomfloat{4}
\def\ext@bottomfloat{lbf}
\def\fnum@bottomfloat{\thebottomfloat}
```

bottomfloat
bottomfloat*
This is the definition of the actual environment. The form with the * is used for double column floats.

```
\newenvironment{bottomfloat}%
                {\@float{bottomfloat}}%
                {\end@float}
\newenvironment{bottomfloat*}%
               {\@dblfloat{bottomfloat}}%
               {\end@dblfloat}
```

9.7.4 Captions

\@makecaption
The \caption command calls \@makecaption to format the caption of floating objects. It gets two arguments, ⟨*number*⟩, the number of the floating object and ⟨*text*⟩, the text of the caption. Usually ⟨*number*⟩ contains a string such as 'Figure 3.2'. The macro can assume it is called inside a \parbox of right width, with \normalsize.

\abovecaptionskip
\belowcaptionskip
These lengths contain the amount of white space to leave above and below the caption.

```
\newlength\abovecaptionskip
\newlength\belowcaptionskip
\setlength\abovecaptionskip{10\p@}
\setlength\belowcaptionskip{10\p@}
```

The definition of this macro is \long in order to allow more then one paragraph in a caption.

```
\long\def\@makecaption#1#2{%
  \vskip\abovecaptionskip
```

We want to see if the caption fits on one line on the page, therefore we first typeset it in a temporary box.

```
  \sbox\@tempboxa{{\captionsize\bfseries #1 -- #2}}%
```

We can the measure its width. It that is larger than the current \hsize we typeset the caption as a centered paragraph.

```
  \ifdim \wd\@tempboxa >\hsize
    {\centering {\captionsize\bfseries #1 -- #2}\par}
```

If the caption fits, we center it. Because this uses an `\hbox` directly in vertical mode, it does not execute the `\everypar` tokens; the only thing that could be needed here is resetting the 'minipage flag' so we do this explicitly.

```
\else
  \global \@minipagefalse
  \hbox to\hsize{\hfil\box\@tempboxa\hfil}%
\fi
\vskip\belowcaptionskip}
```

`\contcaption` The `\contcaption` command can be used to put a 'continuation' caption into a float. It neither increments the float number nor makes any entry in the toc listings.

It is called as `\contcaption{`⟨*continued/concluded*⟩`}{`⟨*optional text*⟩`}`

```
\newcommand{\contcaption}{\@contcaption\@captype}
```

`\@contcaption` This does the work for us.

```
\long\def\@contcaption#1#2{%
   \begingroup
       \@parboxrestore
       \normalsize
       \@makecaption{\csname fnum@#1\endcsname}{\ignorespaces #2}\par
   \endgroup}
```

9.8 Font changing

Here we supply the declarative font changing commands that were common in LaTeX version 2.09 and earlier. These commands work in text mode *and* in math mode. They are provided for compatibility, but one should start using the `\text...` and `\math...` commands instead. These commands are defined using `\DeclareTextFontCommand`, a command with three arguments: the user command to be defined; LaTeX commands to execute in text mode and LaTeX commands to execute in math mode.

`\rm` `\tt` `\sf` The commands to change the family. When in compatibility mode we select the 'default' font first, to get LaTeX2.09 behviour.

```
\DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}
```

`\bf` The command to change to the bold series. One should use `\mdseries` to explicitly switch back to medium series.

```
\DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
```

`\sl` `\it` `\sc` And the commands to change the shape of the font. The slanted and small caps shapes are not available by default as math alphabets, so those changes do nothing in math mode. However, we do warn the user that the selection will not have any effect.One should use `\upshape` to explicitly change back to the upright shape.

```
\DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
\DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
\DeclareOldFontCommand{\sc}{\normalfont\scshape}{\@nomath\sc}
```

`\cal` `\mit` The commands `\cal` and `\mit` should only be used in math mode, outside math mode they have no effect. Currently the New Font Selection Scheme defines these commands to generate warning messages. Therefore we have to define them 'by hand'.

```
\DeclareRobustCommand*{\cal}{\@fontswitch{\relax}{\mathcal}}
\DeclareRobustCommand*{\mit}{\@fontswitch{\relax}{\mathnormal}}
```

9.9 Urls, etc

ISO uses its own format for typesetting urls. This is implemented here via the `url` package.

`\url` `\isourl` The `\url{`⟨*text*⟩`}` command is provided by the `url` package. It may be used for typesetting email addresses. The `\isourl{`⟨*text*⟩`}` command typesets ⟨*text*⟩ in the format required by ISO for an url; that is, the address is underlined and enclosed within (not-underlined) angle brackets.

NOTE: The underlining prohibits linebreaking in the url. I also tried the `ulem` package's `\uline` command, but this also prevented any linebreaking, so we might as well stick to the TEX `\underline`.

```
%%\newcommand{\isourl}[1]{\texttt{<}\underline{\url{#1}}\texttt{>}}
\newcommand{\isourl}[1]{\texttt{<}\url{#1}\texttt{>}}
```

10 Cross Referencing

10.1 Label referencing

`\aref` `\bref` `\cref` `\eref` `\fref` `\nref` `\tref` `\pref` Named references to labeled elements. `\bref{`⟨*label id*⟩`}` is a reference to a labeled informative bibliographic element (similar to the standard LATEX `\cite` command. The others are to named elements of the document.

```
\newcommand{\aref}[1]{\annexrefname~\ref{#1}}
\newcommand{\bref}[1]{[\ref{#1}]}
\newcommand{\cref}[1]{\clauserefname~\ref{#1}}
\newcommand{\eref}[1]{\examplerefname~\ref{#1}}
\newcommand{\fref}[1]{\figurerefname~\ref{#1}}
\newcommand{\nref}[1]{\noterefname~\ref{#1}}
\newcommand{\tref}[1]{\tablerefname~\ref{#1}}
\newcommand{\pref}[1]{\pagerefname~\pageref{#1}}
```

10.2 Table of Contents, etc.

A `\section` command writes a `\contentsline{section}{`⟨*title*⟩`}{`⟨*page*⟩`}` command on the `.toc` file, where ⟨*title*⟩ contains the contents of the entry and ⟨*page*⟩ is the page number. If sections are being numbered, then ⟨*title*⟩ will be of the

form `\numberline{`⟨*num*⟩`}{`⟨*heading*⟩`}` where ⟨*num*⟩ is the number produced by `\thesection`. Other sectioning commands work similarly.

A `\caption` command in a 'figure' environment writes

`\contentsline{figure}{\numberline{`⟨*num*⟩`}{ `⟨*caption*⟩`}}{`⟨*page*⟩`}`

on the `.lof` file, where ⟨*num*⟩ is the number produced by `\thefigure` and ⟨*caption*⟩ is the figure caption. It works similarly for a 'table' environment.

The command `\contentsline{`⟨*name*⟩`}` expands to `\l@`⟨*name*⟩. So, to specify the table of contents, we must define `\l@chapter`, `\l@section`, `\l@subsection`, ... ; to specify the list of figures, we must define `\l@figure`; and so on. Most of these can be defined with the `\@dottedtocline` command, which works as follows.

`\@dottedtocline{`⟨*level*⟩`}{`⟨*indent*⟩`}{`⟨*numwidth*⟩`}{`⟨*title*⟩`}{`⟨*page*⟩`}`

⟨***level***⟩ An entry is produced only if⟨ *level*⟩ <= value of the *tocdepth* counter. Note, `\chapter` is level 0, `\section` is level 1, etc.

⟨***indent***⟩ The indentation from the outer left margin of the start of the contents line.

⟨***numwidth***⟩ The width of a box in which the section number is to go, if ⟨*title*⟩ includes a `\numberline` command.

`\@pnumwidth` `\@tocrmarg` `\@dotsep`

This command uses the following three parameters, which are set with a `\newcommand` (so em's can be used to make them depend upon the font).

`\@pnumwidth` The width of a box in which the page number is put.

`\@tocrmarg` The right margin for multiple line entries. One wants `\@tocrmarg` ≥ `\@pnumwidth`

`\@dotsep` Separation between dots, in mu units. Should be defined as a number like 2 or 1.7

```
\newcommand{\@pnumwidth}{1.55em}
\newcommand{\@tocrmarg} {2.55em}
\newcommand{\@dotsep}{4.5}
```

`\tocentryskip` `\tocbaseline` `\tocskip`

We define two lengths and a utility command.

```
\newlength{\tocentryskip} \setlength{\tocentryskip}{1em}
\newlength{\tocbaseline} \setlength{\tocbaseline}{20pt}
\newcommand{\tocskip}[1]{%
    \addtocontents{toc}{\protect\vspace{#1}}}
```

10.2.1 Table of Contents

`\tableofcontents` This macro is used to request that LaTeX produces a table of contents. In this class the tables of contents, figures etc. are always set in single-column style.

```
\newcommand{\tableofcontents}{%
    \if@twocolumn
      \@restonecoltrue\onecolumn
```

```
    \else
      \@restonecolfalse
    \fi
```

If the document is copyrighted, then the copyright notice is placed at the foot of page ii.

```
%%%     \setcounter{page}{2}
%%%     \thispagestyle{startpage}
%%%     \mbox{}
%%%     \ifc@pyright\@copyrighttext\fi
```

Set the title for the toc, which must start on page (iii) of the document. The actual table of contents is made by calling `\@starttoc{toc}`.

```
%%%     \cleardoublepage
    \setcounter{page}{3}
    \pagestyle{headings}
    \hbox to \textwidth{{\Cfont \contentsname}\hfil\pagename}
```

Add a locator for a bookmark.

```
    \ifisohyper
      \pdfbookmark[1]{\contentsname}{isotoc}%
    \fi
    \begingroup
        \parskip\z@
        \@starttoc{toc}
    \endgroup
```

Finish by restoring two column mode if necessary.

```
    \if@restonecol\twocolumn\fi}
```

Each sectioning command needs an additional macro to format its entry in the table of contents, as described above. In this class the formatting depends on whether or not the `sect` option is used.

`\l@clause` `\l@sclause` `\l@ssclause` `\l@sssclause` `\l@ssssclause` `\l@sssssclause` `\l@annex`

First the default specifications.

```
\newcommand{\l@clause}{\@dottedtocline{1}{0em}{2.3em}}
\newcommand{\l@sclause}{\@dottedtocline{2}{1.5em}{3.2em}}
\newcommand{\l@ssclause}{\@dottedtocline{3}{3em}{4.1em}}
\newcommand{\l@sssclause}{\@dottedtocline{4}{4.5em}{5em}}
\newcommand{\l@ssssclause}{\@dottedtocline{5}{6em}{5.9em}}
\newcommand{\l@sssssclause}{\@dottedtocline{6}{7.5em}{6.8em}}
\newcommand{\l@annex}{\@dottedtocline{1}{0em}{11.0em}}
```

In this class lists of floats are made to appear as though they were an integral part of the table of contents. Further, headings are only printed if there is at least one float of the given kind in the body of the document.

`\floatlist` For print a heading for a list of floats.

```
\newcommand{\floatlist}[2]{%
    \vspace{2\tocentryskip}
```

```
    \hbox to \textwidth{\bfseries #1\hfil}
    \vspace*{\tocentryskip}
    \nopagebreak
    \begingroup
        \parskip\z@
        \@starttoc{#2}
    \endgroup}
```

10.2.2 List of figures

`\iffigs` A flag for figure floats.

```
\newif\iffigs\figsfalse
```

`\listoffigures` This macro is used to request that LaTeX produces a list of figures.

```
\newcommand{\listoffigures}{%
    \ifx\undefined\setfigs\else\setfigs\fi}
```

`\loftnumberline` Used to add a dash after a figure/table number in the listing.

```
\newcommand{\loftnumberline}[1]{#1 --- }

```

`\l@figure` This macro produces an entry in the list of figures. Note that `Figure␣M.999␣` is 6.15em.

```
\newcommand{\l@figure}{\@dottedtocline{1}{0em}{7.5em}}
\renewcommand{\l@figure}[2]{%
  \vskip \z@ \@plus.2\p@
  {%
   \leftskip 0em
   \rightskip \@tocrmarg
   \parfillskip -\rightskip
   \parindent 0em\@afterindenttrue
   \interlinepenalty\@M
   \leavevmode
   \@tempdima 3.15em
   \advance\leftskip \@tempdima \null\nobreak\hskip -\leftskip
   {\let\numberline\loftnumberline \normalfont\figurename{} #1}\nobreak
   \loftfillnum{#2}}
}

\newcommand{\loftfillnum}[1]{\normalfont%
  {\leaders\hbox{$\m@th\mkern 4.5mu\hbox{.}\mkern 4.5mu$}\hfill}\nobreak
  \hb@xt@\@pnumwidth{\hfil #1}\par}


```

10.2.3 List of tables

`\iftabs` A flag for table floats.

```
\newif\iftabs\tabsfalse
```

`\listoftables` This macro is used to request that LaTeX produces a list of tables. It is very similar to `\listoffigures`. Note that `Table␣M.999␣` is 5.75em.

```
\newcommand{\listoftables}{%
    \ifx\undefined\settabs\else\settabs\fi}
```

`\l@table` This macro produces an entry in the list of tables.

```
\newcommand{\l@table}{\@dottedtocline{1}{0em}{6.5em}}

\renewcommand{\l@table}[2]{%
  \vskip \z@ \@plus.2\p@
  {%
   \leftskip 0em
   \rightskip \@tocrmarg
   \parfillskip -\rightskip
   \parindent 0em\@afterindenttrue
   \interlinepenalty\@M
   \leavevmode
   \@tempdima 2.75em
   \advance\leftskip \@tempdima \null\nobreak\hskip -\leftskip
   {\let\numberline\loftnumberline \normalfont\tablename{} #1}\nobreak
   \loftfillnum{#2}}
}

```

`\@caption` This is a reimplementation of the kernel `\@caption` macro (ltfloat.dtx) to cater for the peculiarity of putting the float name before the number in the List of...

```
\long\def\@isocaption#1[#2]#3{%
  \par
  \addcontentsline{\csname ext@#1\endcsname}{#1}%
    {\protect\numberline{{\@nameuse{#1name}} {\@nameuse{the#1}} --- }%
    {\ignorespaces #2}}%
  \begingroup
    \@parboxrestore
    \if@minipage
      \@setminipage
    \fi
    \normalsize
    \@makecaption{\csname fnum@#1\endcsname}{\ignorespaces #3}\par
  \endgroup}

```

10.2.4 ToC and clause numbering

Commands are provided, based on the tocvsec2 package, for changing the section numbering level and the ToC entry level.

`\if@knownclause` `\@setclcnt` Helper macro to set a sectioning-related counter. Use as `\@setclcnt{`⟨*sec*⟩`}{`⟨*counter*⟩`}` to set *counter* to the level of ⟨*sec*⟩.

```
\newif\if@knownclause
\newcommand{\@setclcnt}[2]{
  \@knownclausefalse
  \if\isostringsequal{#1}{none}
    \setcounter{#2}{-10}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{clause}
    \setcounter{#2}{1}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{sclause}
    \setcounter{#2}{2}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{ssclause}
    \setcounter{#2}{3}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{sssclause}
    \setcounter{#2}{4}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{ssssclause}
    \setcounter{#2}{5}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{sssssclause}
    \setcounter{#2}{6}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{all}
    \setcounter{#2}{50}
    \@knownclausetrue
  \fi
  \if@knownclause\else
    \ClassError{isov2}{%
      Unknown clause command name (#1)
    }{%
      I'll ignore it. Type \space <return> and I'll continue.\MessageBreak
      If you haven't mistyped the name then use \protect\setcounter\space instead.}
  \fi
}
```

\settocdepth `\settocdepth{`⟨*sec*⟩`}` is the user command for setting *tocdepth* in the `.toc` file to the value corresponding to ⟨*sec*⟩. It can only be used after the preamble.

```
\newcommand{\settocdepth}[1]{%
  \@knownclausefalse
  \if\isostringsequal{#1}{none}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{-10}}
```

```
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{clause}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{1}}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{sclause}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{2}}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{ssclause}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{3}}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{sssclause}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{4}}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{ssssclause}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{5}}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{sssssclause}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{6}}
    \@knownclausetrue
  \fi
  \if\isostringsequal{#1}{all}
    \addtocontents{toc}{\protect\setcounter{tocdepth}{50}}
    \@knownclausetrue
  \fi
  \if@knownclause\else
    \ClassError{isov2}{%
      Unknown clause command name (#1)
    }{%
      I'll ignore it. Type \space <return> and I'll continue.}
  \fi
}
```

`\maxtocdepth` `\maxtocdepth{`⟨*sec*⟩`}` can be used to initialise *tocdepth* to the value corresponding to ⟨*sec*⟩. This can only be used between the end of the preamble and the `\tableofcontents` command.

```
\newcommand{\maxtocdepth}[1]{%
  \@setclcnt{#1}{tocdepth}
}
```

`\setsecnumdepth` `\setsecnumdepth{`⟨*sec*⟩`}` is the user command for setting *secnumdepth* to the value for ⟨*sec*⟩. It can only be used after the preamble.

```
\newcommand{\setsecnumdepth}[1]{\leavevmode%
  \@setclcnt{#1}{secnumdepth}
}
```

\maxsecnumdepth `\maxsecnumdepth{⟨sec⟩}` can be used to initialise *secnumdepth* after the preamble to the value corresponding to ⟨*sec*⟩.

```
\newcommand{\maxsecnumdepth}[1]{%
  \@setclcnt{#1}{secnumdepth}
}
```

10.3 Bibliography

This class does not implement a bibliography. The `references` environment is defined instead.

10.4 The index

`theindex` The environment 'theindex' can be used for indices. It makes an index with one column, with each entry a separate paragraph. At the user level the commands `\item`, `\subitem` and `\subsubitem` are used to produce index entries of various levels. When a new letter of the alphabet is encountered an amount of `\indexspace` white space can be added.

ISO requires that an index, if present, must be the last element in the document.

```
\newenvironment{theindex}%
  {\clearpage
   \typeout{Index}%
   \refstepcounter{clause}%
   \tocskip{\tocentryskip}%
   \addcontentsline{toc}{index}{\indexname}%
   \columnseprule \z@
   \onecolumn{\fibicl@use*{\indexname}}
   \parindent\z@
   \parskip\z@ \@plus .3\p@\relax
   \let\item\@idxitem}%
  {\clearpage}
```

`\l@index` Format the index entry in the table of contents.

```
\newcommand{\l@index}{\@dottedtocline{1}{0em}{0pt}}
```

`\@idxitem` Thsee macros are used to format the entries in the index.
`\subitem`
`\subsubitem`

```
\newcommand{\@idxitem}  {\par\hangindent 40\p@}
\newcommand{\subitem}   {\par\hangindent 40\p@ \hspace*{20\p@}}
\newcommand{\subsubitem}{\par\hangindent 40\p@ \hspace*{30\p@}}
```

`\indexspace` The amount of white space that is inserted between 'letter blocks' in the index.

```
\newcommand{\indexspace}{\par \vskip 10\p@ \@plus5\p@ \@minus3\p@\relax}
```

The program GenIndex, written for processing ISO documents, takes an `.idx` file and converts it to a `theindex` format. The following are the formatting commands output by GenIndex.

\indexfill
\sindexfill
\ssindexfill

Thse define the format of leaders between the (sub-) topic and the page number. ISO requires a dotted line between each index entry and the page number.

```
\newcommand{\indexfill}{\dotfill}
\newcommand{\sindexfill}{\dotfill}
\newcommand{\ssindexfill}{\dotfill}
```

\indexsee
\indexseealso

These format entries of type 'see ...' and 'see also ...'.

```
\newcommand{\indexsee}[1]{\par \hspace*{2em} \emph{see} #1}
\newcommand{\indexseealso}[1]{\par \hspace*{2em} \emph{see also} #1}
```

\alphaindexspace
\otherindexspace

These format the space between each alphabetic block of entries, and correspondingly for entries that begin with an analpahbetic character. ISO requires no additional spacing.

These commands take one parameter, intended to be the (letter) heading for the next block of entries. For example, we could have defined:

```
\newcommand{\alphaindexspace}[1]{\indexspace
                                 {\bfseries #1}}
```

for printing a vertical space and a bold heading.

```
\newcommand{\alphaindexspace}[1]{}
\newcommand{\otherindexspace}[1]{}
```

For good measure we provide a style file for users of the MAKEINDEX program.

```
⟨/iso⟩
⟨*ist⟩
    %%%%iso.ist  Makeindex style file for ISO documents
group_skip "\n\n"       % no vertical space between blocks
headings_flag 0         % make sure headings are turned off
delim_0 " \\dotfill "   % dot leaders between entry and page numbers
delim_1 " \\dotfill "
delim_2 " \\dotfill "

⟨/ist⟩
⟨*iso⟩
```

10.5 Footnotes

\footnoterule

Usually, footnotes are separated from the main body of the text by a small rule. This rule is drawn by the macro `\footnoterule`. We have to make sure that the rule takes no vertical space (see `plain.tex`) so we compensate for the natural heigth of the rule of 0.4pt by adding the right amount of vertical skip.

To prevent the rule from colliding with the footnote we first add a little negative vertical skip, then we put the rule and make sure we end up at the same point where we begun this operation.

```
\renewcommand{\footnoterule}{%
  \kern-3\p@
```

```
\hrule width .4\columnwidth
\kern 2.6\p@}
```

`\c@footnote` Footnotes are numbered sequentially throughout the document. ISO requires footnotes to be a superscripted arabic numeral with a right parenthesis. The counter is predefined.

```
% \newcounter{footnote}
\renewcommand{\thefootnote}{\arabic{footnote})}
```

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

An example of what can be achieved is given by the following piece of TeX code.

```
\long\def\@makefntext#1{%
   \@setpar{\@@par
            \@tempdima = \hsize
            \advance\@tempdima-10pt
            \parshape \@ne 10pt \@tempdima}%
   \par
   \parindent 1em\noindent
   \hbox to \z@{\hss\@makefnmark}#1}
```

The effect of this definition is that all lines of the footnote are indented by 10pt, while the first line of a new paragraph is indented by 1em. To change these dimensions, just substitute the desired value for '10pt' (in both places) or '1em'. The mark is flushright against the footnote.

In this document class we use a simpler macro, in which the footnote text is set like an ordinary text paragraph, with no indentation except on the first line of a paragraph, and the first line of the footnote. Thus, all the macro must do is set `\parindent` to the appropriate value for succeeding paragraphs and put the proper indentation before the mark.

```
\long\def\@makefntext#1{%
    \parindent 1em%
    \noindent
    \hbox to 1.8em{\hss\@makefnmark}#1}
```

`\@makefnmark` The footnote markers that are printed in the text to point to the footnotes should be produced by the macro `\@makefnmark`. We use the default definition for it.

```
%\def\@makefnmark{\hbox{$^{\@thefnmark}\m@th$}}
```

11 Version control tools

When preparing an international standard the document goes through several iterations. In particular it may change due to international ballot comments. The commands provided may be used to identify changes made to a document during its life cycle.

11.1 Print control

Members of the development group often need to see the changes between document versions, while the general public does not.

`\ifchangemarks` This controls the appearence of the version controls defined below.

```
\newif\ifchangemarks\changemarksfalse
```

The version controls only work properly when the `draft` option is in effect. Also, the command `\changemarkstrue` must be put in the document preamble.

`\v@rid` This acts as an alias for `\marginpar` when both `changemarks` is true and the `draft` option is in effect, otherwise it throws away its two arguments.

```
\newcommand{\v@rid}[2]{%
    \ifchangemarks
       \ifdr@ftd@c
           \marginpar[#1]{#2}%
    \fi\fi}
```

11.2 Change marking

The following commands flag changes in the typeset document. Each of the commands takes one parameter which is intended to be a 'change number' for tracking purposes. Some also take a text parameter which is the changed text.

`\editorial` `\editorial{`⟨*change id*⟩`}` Places the ⟨*change id*⟩ in the document to indicate an editorial change.

```
\newcommand{\editorial}[1]{%
  \@bsphack
    \ifchangemarks
        \v@rid{\small\hfill$^{#1}$ED}%
        {\small ED$^{#1}$\hfill}%
    \fi\@esphack}
```

`\added` `\added{`⟨*text*⟩`}{`⟨*change id*⟩`}` Flags the additional ⟨*text*⟩ with the ⟨*change id*⟩.

```
\long\def\added#1#2{%
  \@bsphack
    \ifchangemarks
        \v@rid{\small\hfill$^{#2}\Rightarrow$}%
        {\small $\Leftarrow^{#2}$\hfill}%
        \emph{#1}%
    \else
```

```
      #1
    \fi\@esphack}
```

`\deleted` `\deleted{`⟨*change id*⟩`}` Places the ⟨*change id*⟩ in the document to indicate that some text has been deleted.

```
\newcommand{\deleted}[1]{%
  \@bsphack
    \ifchangemarks
        \v@rid{\small\hfill$^{#1}\Leftarrow$}%
        {\small $\Rightarrow^{#1}$\hfill}%
    \fi\@esphack}
```

`\moved` `\moved{`⟨*text*⟩`}{`⟨*change id*⟩`}` Flags the moved ⟨*text*⟩ with the ⟨*change id*⟩.

```
\long\def\moved#1#2{%
  \@bsphack
    \ifchangemarks
        \v@rid{\small\hfill$^{#2}\Leftrightarrow$}%
        {\small $\Leftrightarrow^{#2}$\hfill}%
        \emph{#1}%
    \else
       #1
    \fi\@esphack}
```

12 Structure and boilerplate

ISO standard documents have certain required elements and boilerplate.

12.1 Structural elements

`foreword` The `foreword` environment initializes the front matter for a standard and starts an unnumbered foreword clause. To ensure that the front matter is set in single column we use an environment.

```
\newenvironment{foreword}%
  {\tableofcontents
   \listoffigures
   \listoftables
   \clearpage
   \if@twocolumn
     \@restonecoltrue\onecolumn
   \else
     \@restonecolfalse
   \fi
   \fibicl@use*{\forewordname}%
%%    \tocskip{\tocentryskip}%
%%    \addcontentsline{toc}{clause}{\forewordname}%
   \ifisohyper
     \pdfbookmark[1]{\forewordname}{isofwd}%
   \fi}%
```

```
  {\if@restonecol\twocolumn\fi}
```

`\@copyrighttext` This command sets up the copyright notice on the first page of the table of contents. The text is set in a `bottomfloat` environment in a small size.

```
\newcommand{\@copyrighttext}{%
  \vfill
%%%    \begin{bottomfloat}[b]
        \begin{small}
        \copyrightnotice
        \end{small}
%%%    \end{bottomfloat}
}
```

`introduction` Starts a new unnumbered introduction clause, the body of which is set in single column, so we use an environment.

```
\newenvironment{introduction}%
  {\clearpage
   \if@twocolumn
     \@restonecoltrue\onecolumn
   \else
     \@restonecolfalse
   \fi
   \fibicl@use*{\introductionname}%
%%    \tocskip{\tocentryskip}
%%    \addcontentsline{toc}{clause}{\introductionname}%
   \ifisohyper
     \pdfbookmark[1]{\introductionname}{isointro}
   \fi}%
  {\if@restonecol\twocolumn\fi}
```

`\scopeclause` Starts a new numbered scope clause. This is given the label `;i1` as it is the first numbered clause.

```
\newcommand{\scopeclause}{\clause{\scopename}\label{;i1}}
```

`\normrefsclause` Starts a new numbered normative references clause. This is given the label `;i2` as it is the second numbered clause.

```
\newcommand{\normrefsclause}{\clause{\normrefsname}\label{;i2}}
```

`\defclause` `\symclause` `\abbclause` `\defsymclause` `\defabbclause` `\symabbclause` `\defsymabbclause` These macros start new clauses for definitions, symbols and abbreviations. ISO allows these to be grouped in various ways, depending on the amount of material in the respective categories. These are each given the label `;i3` as one should be the third numbered clause.

```
\newcommand{\defclause}{\clause{\defname}\label{;i3}}
\newcommand{\symclause}{\clause{\symname}\label{;i3}}
\newcommand{\abbclause}{\clause{\abbname}\label{;i3}}
\newcommand{\defsymclause}{\clause{\defsymname}\label{;i3}}
\newcommand{\defabbclause}{\clause{\defabbname}\label{;i3}}
\newcommand{\symabbclause}{\clause{\symabbname}\label{;i3}}
\newcommand{\defsymabbclause}{\clause{\defsymabbname}\label{;i3}}
```

\defsubclause These macros start new sub-clauses for definitions, symbols and abbreviations.
\symsubclause ISO allows these to be grouped in various ways, depending on the amount of
\abbsubclause material in the respective categories.

```
\defsymsubclause
\defabbsubclause
\symabbsubclause
```

```
\newcommand{\defsubclause}{\sclause{\defname}}
\newcommand{\symsubclause}{\sclause{\symname}}
\newcommand{\abbsubclause}{\sclause{\abbname}}
\newcommand{\defsymsubclause}{\sclause{\defsymname}}
\newcommand{\defabbsubclause}{\sclause{\defabbname}}
\newcommand{\symabbsubclause}{\sclause{\symabbname}}
```

\fcandaclause This macro starts a clause 'Fundamental concepts and assumptions'. The actual title is given by the value of `\fcandaname`.

```
\newcommand{\fcandaclause}{\clause{\fcandaname}}
```

\bibannex This macro starts a bibliography (which used to be an informative annex).

```
\newcommand{\bibannex}{%
  \typeout{Bibliography}
  \clearpage
  \fibicl@use*{\bibname}
  \tocskip{\tocentryskip}
  \addcontentsline{toc}{index}{\bibname}
}
```

12.2 Boilerplate

ISO defines the wording of certain textual elements within a standard.

This class has been prepared for standard documents in the English language. The boilerplate text commands must be redefined for other languages.

\copyrightnotice The required English text of the copyright notice.

```
\newcommand{\copyrightnotice}{%
\copyright\quad \copyrightname\quad \thesyear\newline
All rights reserved. Unless otherwise specified, no part of
this publication may be reproduced or utilized in any form or
by any means, electronic or mechanical, including photocopying
and microfilm, without permission in writing from %%% the publisher.
%%%\makebox[\textwidth][r]{%
%%%ISO/IEC Copyright Office $\bullet$ Case Postale 56 $\bullet$
%%%CH-1211 Gen{\'e}ve 20 $\bullet$ Switzerland}
%%\vspace{\baselineskip}\newline
%%\hspace*{1em} International Organization for Standardization\newline
%%\hspace*{1em} Case Postale 56 $\bullet$ CH-2111 Gen{\'e}ve 20 $\bullet$ Switzerland
either ISO at the address below or ISO's member body in the country
of the requester.
\par
\noindent ISO copyright office \\
Case postale 56. CH-1211 Geneva 20 \\
Tel. +41 22 749 01 11 \\
```

```
Fax  +41 22 734 10 79 \\
E-mail \texttt{copyright@iso.ch} \\
Web \texttt{www.iso.ch}
```

For an IS or a Tech Report, need a blank line and place of printing

```
%%\ifisst@ndard \ifc@pyright
%%    \vspace{\baselineskip}\newline\noindent
%%    Printed in Switzerland
%%\fi\fi
%%\ift@chrep \ifc@pyright
%%    \vspace{\baselineskip}\newline\noindent
%%    Printed in Switzerland
%%\fi\fi
}
```

`\fwdbp` The prescribed text of the initial paragraphs in an ISO Standard Foreword.

```
\newcommand{\fwdbp}{\input{isofwdbp}}
```

The following is the text contained in the file `isofwdbp.tex`.

```
⟨/iso⟩
⟨*fwd1⟩
\ProvidesFile{isofwdbp.tex}[2001/08/29 Boilerplate for start of Foreword]

ISO (the International Organization for Standardization) is a worldwide
federation of national standards bodies (ISO member bodies). The work
of preparing International Standards is normally carried out through
ISO technical committees. Each member body interested in a subject for
which a technical committee has been established has the right to be
represented on that committee. International organizations,
governmental and non-governmental, in liaison with ISO, also take part
in the work. ISO collaborates closely with the International
Electrotechnical Commission (IEC) on all matters of electrotechnical
standardization.

International Standards are drafted in accordance with the rules given
in the ISO/IEC Directives, Part~2.

The main task of technical committees is to prepare International Standards.
Draft International Standards adopted by the technical committees are
circulated to the member bodies for voting. Publication as an
International Standard requires approval by at least 75\% of the member
bodies casting a vote.
\par

⟨/fwd1⟩
⟨*iso⟩
```

`\tspasfwdbp` The prescribed text of the initial paragraphs in an ISO Technical Specification or PAS Foreword.

```
\newcommand{\tspasfwdbp}{\input{tspasfwdbp}}
```

The following is the text contained in the file `tspasfwdbp.tex`.

```
⟨/iso⟩
⟨*tspasfwd1⟩
\ProvidesFile{tspasfwdbp.tex}[2001/07/06 Boilerplate for start of TS/PAS Foreword]

ISO (the International Organization for Standardization) is a worldwide
federation of national standards bodies (ISO member bodies). The work
of preparing International Standards is normally carried out through
ISO technical committees. Each member body interested in a subject for
which a technical committee has been established has the right to be
represented on that committee. International organizations,
governmental and non-governmental, in liaison with ISO, also take part
in the work. ISO collaborates closely with the International
Electrotechnical Commission (IEC) on all matters of electrotechnical
standardization.

International Standards are drafted in accordance with the rules given
in the ISO/IEC Directives, Part~2.

    The main task of technical committees is to prepare International
Standards.
Draft International Standards adopted by the technical committees are
circulated to the member bodies for voting. Publication as an
International Standard requires approval by at least 75\% of the member
bodies casting a vote.

    In other circumstances, particularly when there is an urgent market
requirement for such documents, a technical committee may decide to
publish other types of normative document:
\begin{itemize}
\item an ISO Publicly Available Specification (ISO/PAS) represents an
agreement between technical experts in an ISO working group and is
accepted for publication if it is approved by more than 50\% of the
members of the parent committee casting a vote;

\item an ISO Technical Specification (ISO/TS) represents an agreement
between the members of a technical committee and is accepted for
publication if it is approved by 2/3 of the members of the committee
casting a vote.
\end{itemize}

    An ISO/PAS or ISO/TS is reviewed every three years with a view to
deciding whether it can be transformed into an International Standard.
\par

⟨/tspasfwd1⟩
```

The following is the text contained in the file `trfwd1.tex`.

```
⟨*trfwd1⟩
   %% trfwd1.tex  Boilerplate for start of a tech rep Foreword clause
```

```
%

    ISO (the International Organization for Standardization) is a worldwide
federation of national standards bodies (ISO member bodies). The work
of preparing International Standards is normally carried out through
ISO technical committees. Each member body interested in a subject for
which a technical committee has been established has the right to be
represented on that committee. International organizations,
governmental and non-governmental, in liaison with ISO, also take part
in the work. ISO collaborates closely with the International
Electrotechnical Commission (IEC) on all matters of electrotechnical
standardization.

    International Standards are drafted in accordance with the rules
given in the ISO/IEC Directives, Part 3.

    The main task of technical committees is to prepare International
Standards. Draft International Standards adopted by the technical
committees are circulated to the member bodies for voting. Publication
as an International Standard requires approval by at least 75\% of the
member bodies casting a vote.

    In other circumstances, particularly when there is an urgent market
requirement for such documents, a technical committee may decide to
publish other types of normative document:
\begin{itemize}
\item an ISO Publicly Available Specification (ISO/PAS) represents an
agreement between technical experts in an ISO working group and is
accepted for publication if it is approved by more than 50\% of the
members of the parent committee casting a vote;

\item an ISO Technical Specification (ISO/TS) represents an agreement
between the members of a technical committee and is accepted for
publication if it is approved by 2/3 of the members of the committee
casting a vote.
\end{itemize}

    An ISO/PAS or ISO/TS is reviewed every three years with a view to
deciding whether it can be transformed into an International Standard.
\par


⟨/trfwd1⟩
⟨*iso⟩
```

`\trfwdbpi` `\trfwdbpii` Required texts for a technical report foreword. Use as: `\trfwdbpii{`⟨*application field*⟩`}`.

```
\newcommand{\trfwdbpi}{\input{trfwd1}}
\newcommand{\trfwdbpii}[1]{%
  \ClassError{iso}{The \protect\trfwdpbii\space command has been removed}%
```

```
    {Type <return> to proceed, and change your source file before
     running LaTeX again.}
}
```

\intropatents `\intropatents` is the boilerplate for the last Introduction paragraph dealing with potential additional patent rights.

```
\newcommand{\intropatents}{\par
  Attention is drawn to the possibility that some of the elements of this
document may be the subject of patent rights
other than those mentioned above.
ISO [and/or] IEC shall not be held responsible
for identifying any or all such patent rights.\par}

```

\fwdnopatents `\fwdnopatents` is the boilerplate for the Foreword paragraph dealing with potential patent rights.

```
\newcommand{\fwdnopatents}{\par
  Attention is drawn to the possibility that some of the elements of this
document may be the subject of patent rights.
ISO shall not be held responsible
for identifying any or all such patent rights.\par}

```

\normrefbp The required text for the introduction of the normative references clause. Use as: `\normrefbp{`⟨*standard identifier*⟩`}`

```
\newcommand{\normrefbp}[1]{%

The following normative documents contain provisions which, through
reference in this text, constitute provisions of this #1.
For dated references, subsequent amendments to, or revisions of,
any of these publications do not apply.
However, parties
to agreements based on this #1
are encouraged to investigate the possibility of applying
the most recent editions of the normative documents indicated below.
For undated references, the latest edition of the normative
document referred to applies.
Members of ISO and IEC maintain registers of currently
valid International Standards.

}
```

13 Initialization

13.1 Words and phrases

\annexname
\bibname
\contentsname
\defname
\symname
\abbname
\defabbname
\defsymname
\defsymabbname
\fcandaname
\forewordname
\indexname

This document class is for documents prepared in the English language. To prepare a version for another language, various English words and phrases must be

replaced. The English elements that require replacement are defined below in command names.

This list is for titles of document sections.

```
\newcommand{\abbname}{Abbreviations}
\newcommand{\annexname}{Annex}
\newcommand{\bibname}{Bibliography}
\newcommand{\contentsname}{Contents}
\newcommand{\defname}{Terms and definitions}
\newcommand{\defabbname}{Terms, definitions, and abbreviations}
\newcommand{\defsymname}{Terms, definitions, and symbols}
\newcommand{\defsymabbname}{Terms, definitions, abbreviations, and symbols}
\newcommand{\fcandaname}{Fundamental concepts and assumptions}
\newcommand{\forewordname}{Foreword}
\newcommand{\indexname}{Index}
\newcommand{\informativename}{informative}
\newcommand{\introductionname}{Introduction}
\newcommand{\normativename}{normative}
\newcommand{\normrefsname}{Normative references}
\newcommand{\scopename}{Scope}
\newcommand{\sectionname}{Section}
\newcommand{\symname}{Symbols}
\newcommand{\symabbname}{Symbols and abbreviations}
```

\copyrightname
\examplename
\figurename
\inscopename
\ISname
\listannexname
\listfigurename
\listtablename
\notename
\outofscopename
\pagename
\tablename
\tbpname

These are the names and phrases used for general elements.

```
\newcommand{\copyrightname}{ISO}
\newcommand{\examplename}{EXAMPLE}
 %%%\newcommand{\examplesname}{EXAMPLES}
\newcommand{\figurename}{Figure}
\newcommand{\inscopename}{The following are within the scope of this }
\newcommand{\ISname}{INTERNATIONAL STANDARD}
\iffdisstandard\renewcommand{\ISname}{FINAL DRAFT INTERNATIONAL STANDARD}\fi
\ifdisstandard\renewcommand{\ISname}{DRAFT INTERNATIONAL STANDARD}\fi
\ifcdstandard\renewcommand{\ISname}{COMMITTEE DRAFT}\fi
\ifwdstandard\renewcommand{\ISname}{WORKING DRAFT}\fi
\iftechrep\renewcommand{\ISname}{TECHNICAL REPORT}\fi
\iftechspec\renewcommand{\ISname}{TECHNICAL SPECIFICATION}\fi
\ifpaspec\renewcommand{\ISname}{PUBLICLY AVAILABLE SPECIFICATION}\fi
\ifotherdoc\renewcommand{\ISname}{}\fi
\newcommand{\listannexname}{Annexes}
\newcommand{\listfigurename}{Figures}
\newcommand{\listtablename}{Tables}
\newcommand{\notename}{NOTE}
 %%%\newcommand{\notesname}{NOTES}
\newcommand{\outofscopename}{The following are outside the scope of this }
\newcommand{\pagename}{Page}
\newcommand{\tablename}{Table}
\newcommand{\tbpname}{To be published.}
```

\annexrefname
\clauserefname
\examplerefname
\figurerefname
\noterefname
\tablerefname
\pagerefname

These are the names for referenced document elements. Except when starting

a sentence or referring to a figure, references to document elements start with a lower case letter.

```
\newcommand{\annexrefname}{annex}
\newcommand{\clauserefname}{clause}
\newcommand{\examplerefname}{example}
\newcommand{\figurerefname}{Figure}
\newcommand{\noterefname}{note}
\newcommand{\tablerefname}{Table}
\newcommand{\pagerefname}{page}
```

`\abstractname` `\appendixname` `\chaptername` `\partname` `\refname`

These names are used in the standard LaTeX classes but are not applicable in this class. We just make them null.

```
\newcommand{\abstractname}{}
\newcommand{\appendixname}{}
\newcommand{\chaptername}{}
\newcommand{\partname}{}
\newcommand{\refname}{}
```

13.2 Date

`\today`

This macro uses the TeX primitives `\month`, `\day` and `\year` to provide the date of the LaTeX-run.

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

13.3 Two column mode

`\columnsep`

This gives the distance between two columns in two column mode.

```
\setlength\columnsep{10\p@}
```

`\columnseprule`

This gives the width of the rule between two columns in two column mode. We have no visible rule.

```
\setlength\columnseprule{0\p@}
```

13.4 The page style

We use the page style *headings* by default and start with roman numbering for the front matter, this being reset to arabic by the title or first main matter section/clause.

```
\pagestyle{headings}
\pagenumbering{roman}
```

We set the sectional counters to zero and the `tocdepth` to one (clauses only listed).

```
\setcounter{clause}{0}
\setcounter{annex}{0}
\setcounter{tocdepth}{1}
```

13.5 Single or double sided printing

We do not try to make each page as long as all the others, even though it is two-side printing.

```
\@twosidetrue
\raggedbottom
```

When the twocolumn option was specified we call `\twocolumn` to activate this mode. We try to make each column as long as the others, but call `sloppy` to make our life easier.

```
\if@twocolumn
  \twocolumn
  \sloppy
  \flushbottom
```

Normally we call `\onecolumn` to initiate typesetting in one column.

```
\else
  \onecolumn
\fi
```

The end of the class definitions.

⟨/iso⟩

14 The askinc package

This package provides an interactive 'include' facility. It was developed by Phil Spiby of CADDETC, Leeds, United Kingdom in the late eighties.

⟨∗inc⟩

`\infile` The `\infile{`⟨*file name*⟩`}` command is a cross between the `\input` and `\include` commands. When this package is used, at runtime the user is asked to interactively specify a comma-seperated list of the names of `\infile`d files that are to be processed. In this sense it acts like the `\include` and `\includeonly` pair of commands. If no list is entered at the terminal (by hitting the ⟨*RETURN*⟩ key) then all `\infile`d files are processed. In this sense it acts like the `\input` command. However, like the `\include` command, an `\infile`d file cannot contain any other `\infile`d file.

`temp` Define a counter *temp* for general use within the include files. This is required to ensure that the contents of `\incfiles` is used and not `\incfiles` the string.

```
\newcounter{temp}
```

Now for the rest of the definition.

```
\def\readinclude#1\endread{\gdef\myincludeonly{\includeonly{#1}}}
\long\def\stripspace#1 \nextspace{#1}
\typeout{Which files do you want processing ?}
\message{enter names (separated by commas) or <RET> for all.}
\message{}\global\read-1 to\incfiles
```

```
\if\incfiles\par\let\infile\input
\else\let\infile\include
\edef\incfiles{\expandafter\stripspace\incfiles\nextspace}
\expandafter\readinclude\incfiles\endread\myincludeonly\fi
```

The end of the askinc package.

⟨/inc⟩

References

[GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley Publishing Company, 1994.

[ISO97] ISO/IEC Directives Part 3. *Drafting and presentation of International Standards*, Third edition, 1997.

[ISO01] ISO/IEC Directives Part 2. *Rules for the structure and drafting of International Standards*, Fourth edition, 2001.

[Wil96] Peter R. Wilson. *LaTeX for standards: The LaTeX package files user manual*. NIST Report NISTIR, June 1996.

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

C

D

J

L

M

N

O