The aeb-minitoc Package

D. P. Story
Email: dpstory@uakron.edu

processed October 3, 2018

Contents

```
⟨*package⟩
```

\ifMiniTocListings

nominitocs

!nominitocs

The `\ifMiniTocListings` is a Boolean switch, which when true signals that there is a non-empty listing; otherwise, it is set to false. It is used to display a latex warning to the user that the listing is empty. Also, globally, `\ifMiniTocListings` is set to false when `nominitocs` is taken. The other option is `!nominitocs` is a convenience option; it is not 'not' version of `nominitocs`; when `!nominitocs` is specified, mini-tocs are created. This is the same as specifiying no option at all.

```
\newif\ifMiniTocListings \MiniTocListingstrue
\DeclareOption{nominitocs}{\AtEndOfPackage{\MiniTocListingsfalse
  \let\insertminitoc\insertminitocNOT}}
\DeclareOption{!nominitocs}{\MiniTocListingstrue}
\ProcessOptions
```

1 Description

A simple mini-toc package; originally designed for web, but now works for all standard LaTeX classes. The main user command is `\insertminitoc`, defined below.

Our approach is to use each entry the `\jobname.toc` as the first argument of the macro `\mtocCL`, a second argument keeps a running count on the number of entries.

```
\mtocCL{\contentsline{section}{\numberline{1}Section Title}{2}}{cnt} or
\mtocCL{\contentsline{section}
   {\numberline{1}Section Title}{2}{section.1}}{cnt}
```

`\contentsline` has four arguments when `hyperref` is loaded and three otherwise. When inserting the full table of contents, we define `\def\mtocCL#1#2{#1}` to do nothing. When we are building a mini-toc, we `\let \mtocCL` to `\mtoc@CL@mtoc`. The effect of this macro is to remove any entry (in `\jobname.toc`) that does not contain `\contentsline` as it first token and to position the `cnt` argument for later use. But by then `\contentsline` has already been `\let` to `\cl@LOOKFORSEC`. Now `\cl@LOOKFORSEC` determines whether any particular entry should be displayed in the current mini-toc.

2 Documentation and Code.

As a demonstration of this package, we present a mini-toc for this section, which only has `\paragraph` and `\subparagraph` section headings.

The verbatim listing for this mini-toc is

```
\TOCLevels{section}{subparagraph}
\begin{minitocfmt}{\minitocFmt}
  \declaretocfmt{paragraph}{\@W{1em}\@D{0em}}
  \declaretocfmt{subparagraph}{\@W{1.5em}\@D{1em}}
\end{minitocfmt}

\begin{center}\minitocFmt
\fbox{\begin{minipage}{0.8\linewidth}\centering
\textbf{Contents of this section}\vadjust{\kern3pt}%
\insertminitoc\relax
\end{minipage}}
\end{center}
```

Contents of this section

We begin by saving the definitions macros we modify later.

```
\let\mtoc@contentsline\contentsline
\let\mtoc@starttoc\@starttoc
\let\mtoc@tableofcontents\tableofcontents
%\let\mtoc@addtocontents\addtocontents
```

`\@minitocCnt` Some counters and utility macros. The counter `\@minitocCnt` is incremented

\mtocgobble in the redefined \addtocontents command. The command \mtocgobble is a 'public' version of the core LaTeX command \@gobble.

```
\newcount\@minitocCnt \@minitocCnt=0\relax
\def\csarg#1#2{\expandafter#1\csname#2\endcsname}
\let\mtoc@One=1 \let\mtoc@Zero=0
\let\mtocgobble\@gobble
```

¶ The top and bottom most. The package assigns the top level and bottom level automatically, based upon the class being used; the document author can override these for the whole document, or for particular mini-tocs. \TOPLevel \TOPLevel{⟨*name*⟩} is the name of the top level. It is expected that a mini-toc will be inserted with each top level in the document, as the author's discretion. \BTMLevel The \BTMLevel{⟨*name*⟩} is the name of the bottom most level. A mini-toc consists of all sections *beneath* the top level and *above* the bottom level. Thus, if \TOPLevel{chapter} and \BTMLevel{subsubsection}, then the mini-toc contans all \section and \subbsection title headings within the current chapter.

\TOPLevel \BTMLevel **¶¶ Manually set the top and bottom levels.** \TOPLevel{⟨*top-level*⟩} and \BTMLevel{⟨*btm-level*⟩} are used to determine what entries are to be included in the mini-toc. As a convenience,

\TOCLevels \TOCLevels{⟨*top-level*⟩}{⟨*btm-level*⟩}

can declare both at once. If an argument is empty, the current level is used.

```
\def\TOPLevel#1{\def\x{#1}\ifx\x\@empty\else
  \def\mtoc@TOPLevel{#1}\edef\TOPLevelNum{\@nameuse{sl@#1}}\fi}
\def\BTMLevel#1{\def\x{#1}\ifx\x\@empty\else
  \def\mtoc@BTMLevel{#1}\edef\BTMLevelNum{\@nameuse{sl@#1}}\fi}
\def\TOCLevels#1#2{\TOPLevel{#1}\BTMLevel{#2}}
```

¶¶ Automatically set the top and bottom levels. We make reasonable choices for book, report, and article; these are the three classes that this package supports. In the course, we define, in macro form, the levels of each of these section names (\sl@⟨*sec-name*⟩ and \sl@⟨*sec-name*⟩*).

```
\@ifclassloaded{book}{%
  \TOCLevels{chapter}{subsection}
  \def\sl@part{-1}\def\sl@chapter{0}
  \csarg{\edef}{sl@part*}{\sl@part}
  \csarg{\edef}{sl@chapter*}{\sl@chapter}
}{%
  \@ifclassloaded{report}{%
    \TOCLevels{chapter}{subsection}
    \def\sl@part{-1}\def\sl@chapter{0}
    \csarg{\edef}{sl@part*}{\sl@part}
    \csarg{\edef}{sl@chapter*}{\sl@chapter}
  }{%
   \TOCLevels{section}{subsubsection}
    \def\sl@part{0}\csarg{\edef}{sl@part*}{\sl@part}
  }%
}
\def\sl@section{1}\def\sl@subsection{2}\def\sl@subsubsection{3}
\def\sl@paragraph{4}\def\sl@subparagraph{5}%\def\sl@all{17}
\csarg{\edef}{sl@section*}{\sl@section}
\csarg{\edef}{sl@subsection*}{\sl@subsection}
\csarg{\edef}{sl@subsubsection*}{\sl@subsubsection}
\csarg{\edef}{sl@paragraph*}{\sl@paragraph}
\csarg{\edef}{sl@subparagraph*}{\sl@subparagraph}
\newif\if@foundTOPLevel \@foundTOPLevelfalse
```

\cl@LOOKFORSEC

\cl@LOOKFORSEC@LX

\cl@LOOKFORSEC@HY

The command \insertminitoc, just before inputting \jobname.toc, \lets \contentsline to \cl@LOOKFORSEC. This command then looks for lines at the top most section level, if it finds one, and the section number matches the one set by \insertminitoc (\mtoc@sec), it sets \if@foundTOPLevel to true, and stores all subsequent lines in \toks@ until another section is encountered, at which time \if@foundTOPLevel is set to false. There are two versions of \cl@LOOKFORSEC: (1) \cl@LOOKFORSEC@LX for when hyperref is not loaded; and (2) \cl@LOOKFORSEC@HY for when hyperref is loaded.

\mtoc@@contentsline takes five arguments, we save the page number (#3) the hyperref anchor (#4) and the TOC entry number (#5). The definitions made within \mtoc@@contentsline are later \let to \@PgNum, \@L, and \@E. We grab #5, which is the entry count, and pass the rest to \mtoc@contentsline.

```
\def\mtoc@@contentsline#1#2#3#4#5{\def\mtoc@PgNum{#3}%
  \def\mtoc@HY@anchor{#4}\def\TOCEntryNum{#5}%
  \mtoc@contentsline{#1}{#2}{#3}{#4}}
```

All but the last argument in both of these next two command are the standard arguments for \contentsline. The last argument is one introduced by this package; it keeps the count of the TOC entries. This last argument is used to identify the top level section.

```
\long\def\cl@LOOKFORSEC@LX#1#2#3#4{%
  \cl@LOOKFORSEC@HY{#1}{#2}{#3}{\@empty}{#4}}
\long\def\cl@LOOKFORSEC@HY#1#2#3#4#5{\def\mt@rgi{#1}% dps
  \if@foundTOPLevel\ifx\mtocCL\@gobbletwo\else
      \edef\NUMLevel{\@nameuse{sl@#1}}%
      \ifnum\NUMLevel > \TOPLevelNum\relax\else
        \let\mtocCL\@gobbletwo
      \fi
  \fi\fi
  \@chkForNl#2\@nil % is it a * section
  \ifx\mtocCL\@gobbletwo\else
    \edef\mtoc@tmp{\the\mtocs@toks@}%
      \ifx\mt@rgi\mtoc@TOPLevel
        \def\SECNUM{MTOC.#5}%
        \set@display@protect
        \edef\mtoc@sec{\mtoc@sec}%
        \ifx\SECNUM\mtoc@sec
            \@foundTOPLeveltrue
        \else
```

```
     \@foundTOPLevelfalse\fi
    \set@typeset@protect
   \else
    \if@foundTOPLevel
     \ifnum\NUMLevel > \BTMLevelNum\relax\else
      \ifx\@nlrtn\relax
       \mtocs@toks@=\expandafter{\mtoc@tmp
       \mtoc@@contentsline{#1}{#2}{#3}{#4}{#5}}\else
       \mtocs@toks@=\expandafter{\mtoc@tmp
       \mtoc@@contentsline{#1*}{#2}{#3}{#4}{#5}}\fi
    \fi
   \fi
  \fi
 \fi
}
```

This version of `\mtoc@BTMLevel` only accepts lines that are not `subsubsection`.

```
\@ifpackageloaded{hyperref}{\let\cl@LOOKFORSEC\cl@LOOKFORSEC@HY}
 {\let\cl@LOOKFORSEC\cl@LOOKFORSEC@LX}
```

`\@chkForNl` determines if the first token is `\numberline`.

```
\def\@chkForNl#1#2\@nil{% check for number line
 \ifx#1\numberline\let\@nlrtn\relax\else
 \def\@nlrtn{\numberline{\hfill}}\fi}
```

¶ Modify `\tableofcontents`

```
\def\mtoc@st@rttoc#1{\begingroup
  \if@filesw \expandafter\newwrite\csname tf@#1\endcsname
  \immediate\openout \csname tf@#1\endcsname \jobname.#1\relax
  \fi\global\@nobreakfalse\endgroup}
\let\mtoc@@starttoc\mtoc@One
\def\mtoc@start@toc{\let\mtoc@start@next\relax
 \@ifundefined{aebLastPage}
 {%
  \ifx\mtoc@@starttoc\mtoc@One
   \global\let\mtoc@@starttoc\mtoc@Zero
   \def\mtoc@start@next{\mtoc@st@rttoc{toc}}\fi
  \mtoc@start@next
 }{%
  \ifnum\aebLastPage<\thepage\relax
   \def\mtoc@start@next{\mtoc@st@rttoc{toc}}\fi
  \mtoc@start@next
 }%
}
\AtEndDocument{\mtoc@start@toc}
```

¶ Modify `\addtocontents`. (`\addtocontents{toc}{`⟨*content*⟩`}`) If the document author inserts vertical spacing, or other formatting, that could be problems in the minitoc. So we'll try to remove it. We begin by placing the second argument ⟨*content*⟩ as the argument of a command, `\mtocCL{`⟨*content*⟩`}`. Initially,

\mtocCL just passes its argument into the TEX stream. Later, it will be redefined within \insertminitoc.

```
\def\mtoc@TOC{toc}
\def\mtocCL#1#2{#1}
```

Here, we modify the macro \addtocontents to insert \mtocCL.

```
\newcommand\addtocontents@mtoc[2]{%
    \def\mt@rgi{#1}\ifx\mt@rgi\mtoc@TOC
    \global\advance\@minitocCnt\@ne
    \mtoc@addtocontents{#1}{\protect\mtocCL{#2}{\the\@minitocCnt}}\else
    \mtoc@addtocontents{#1}{#2}\fi}
\AtBeginDocument{\let\mtoc@addtocontents\addtocontents
  \let\addtocontents\addtocontents@mtoc}
```

Modify the \tableofcontents to \mtoc@tableofcontents. We compensate later by executing \mtoc@start@toc at the end of the document.

```
\def\tableofcontents{%
    \def\@starttoc##1{\makeatletter
    \@input{\jobname.##1}\makeatother}%
    \NoFmtTOCEntry
    \mtoc@tableofcontents
    \global\let\@starttoc\mtoc@starttoc}
```

\mtoc@CL@mtoc

\mtoc@CL@mtoc is the redefined version of \mtocCL, as described above. We attempt to see if the first token of its argument is \contentsline, if yes we pass it on, otherwise, we gobble it.

```
\newcommand{\mtoc@CL@mtoc}[1]{\mtoc@parse#1\@nil}
\def\mtoc@parse#1#2\@nil{\ifx#1\contentsline
  \def\mtoc@next##1{#1#2{##1}}\else
  \let\mtoc@next\@gobble\fi\mtoc@next}
```

¶ Modify \@startsection and referencing. We redefine \@startsection to pick up the first argument (the section name) and define \@currentsecname, which is use in a simple cross referencing system needed for this mini-toc package. This package should be loaded after hyperref for sure.

```
\let\@startsection@mtoc@SAVE\@startsection
\def\@startsection#1{\def\@currentsecname{#1}%
  \@startsection@mtoc@SAVE{#1}}
```

\mtoclabel
\mtocref

The use of \mtoclabel and \mtocref are not needed unless you redefine a section heading to a non-numerical value. This system needs a section number.

```
\def\mtoclabel#1{\label{#1}\@bsphack
    \protected@write\@auxout{}{\string
    \csarg{\string\gdef}{mtoclbl#1}{\the\@minitocCnt}
  }%
  \@esphack
}
\def\mtocref#1{\@nameuse{mtoclbl#1}}
```

`\insertminitoc` ¶ `\insertminitoc`: **The main command.** `\insertminitoc` is the main user command for this package, it places a "minitoc" for a section (`\mtoc@TOPLevel`) of a document, listing only the subsections within that section. It takes an optional argument for indicating the section number, the subsections of which are to be displayed. The default is the current section, `\@nameuse{the\mtoc@TOPLevel}`.

`\if@minitoc` This Boolean is set to true, in a group, when `\insertminitoc` is expanded. This is to support a feature for formatting a mini-toc entry; `\miniorfulltoc` is used for this purpose. `\miniorfulltoc` is inserted in the optional argument of a section command:

```
\subsection[\protect
  \miniorfulltoc{\textbf}{Subsection Entry}]{Subsection Entry}
```

`\miniorfulltoc{`⟨*fmt*⟩`}{`⟨*entry*⟩`}` The first argument of `\miniorfulltoc` is passed to the second entry; for example, `{\textbf{Subsection Entry}}` as an argument and in a group. Thus, the first argument can be a command with one argument, or a command with not arguments.

```
\newif\if@minitoc \@minitocfalse
\newif\if@MiniTocListings \@MiniTocListingstrue
\def\NoFmtTOCEntry{\@minitocfalse}
\def\FmtTOCEntry{\@minitoctrue}
\def\miniorfulltoc#1#2{\if@minitoc
  {{#1{#2}}}\else{#2}\fi}
```

`\insertminitoc[`⟨*label-name*⟩`]` After the above preliminaries, we get to `\insertminitoc`. The default value of the optional parameter is `MTOC.\the\@minitocCnt`; thus, we use the most recent value of `\@minitocCnt`. An explicit argument is needed when the mini-toc is placed somewhere else (after `\minitocCnt` has been incremented). You can also say `\insertminitoc[`⟨*label-name*⟩`]`, where ⟨*label-name*⟩ is a label name set by the `\mtoclabel` command.

```
\newcommand{\insertminitoc}[1][]{%
  \def\mtoc@rgi{#1}\ifx\mtoc@rgi\@empty
    \edef\mtoc@rgi{MTOC.\the\@minitocCnt}\else
    \edef\mtoc@rgi{MTOC.\mtocref{#1}}\fi
  \ifnum\TOPLevelNum > \BTMLevelNum
    \PackageError{aeb-minitoc}{%
      The top level (\mtoc@TOPLevel) must be\MessageBreak
      must be higher on the hierarchy then at bottom level}
      {Try switching the two}\fi
  \begingroup
  \edef\mtoc@sec{\mtoc@rgi}\mtocs@toks@={}%
```

`\let \contentsline` to `\cl@LOOKFORSEC`

```
  \let\contentsline\cl@LOOKFORSEC
```

`\let \mtocCL` to `\mtoc@CL@mtoc`

```
  \let\mtocCL\mtoc@CL@mtoc
  \@foundTOPLevelfalse
  \let\mtoc@numberline\numberline
```

Insert formatting (\Pg = \sl@@sNumFmt) for the page number here.

```
\def\numberline##1{\makebox[\mtoc@numBoxWidth][l]%
  {\sl@@sNumFmt{##1}}\sl@@EntryFmt}%
\makeatletter\InputIfFileExists{\jobname.toc}%
{\PackageInfo{aeb-minitoc}{TOC file read}}
{\PackageInfo{aeb-minitoc}{TOC file not available}}%
\edef\x{\the\mtocs@toks@}\ifx\x\@empty
  \global\@MiniTocListingsfalse\else
  \global\@MiniTocListingstrue\fi
```

Insertion point. This is where the mini-toc entries are entered into the latex stream to be typeset.

```
\the\mtocs@toks@\par\makeatother
\if@MiniTocListings\else
\PackageWarning{aeb-mintoc}{No mini-toc built here}\fi
\endgroup
}
```

\insertminitocNOT

When the nominitocs option is in effect, we \let the command \insertminitoc to \insertminitocNOT, which absorbs all its arguments.

```
\newcommand{\insertminitocNOT}[1][]{}
```

\numBoxWidth{⟨*length*⟩}

The \mtoc@numBoxWidth determines the width of the \hbox that contains the section number. It is conveniently set using \numBoxWidth. The default declaration is \numBoxWidth{2.5em}. The ⟨*length*⟩ should be measured in em units. Within the minitocfmt, \@W is \let to \numBoxWidth.

```
\def\numBoxWidth#1{\def\mtoc@numBoxWidth{#1}}
\numBoxWidth{2.5em}
```

In its “raw” expansion, \insertminitoc may not be what you want; in this case, enclose it in some appropriate environment. The following is an example of how to use this command. This can be part of a command that inserts code just after every \section.

```
\begin{center}\minitocFmt
\begin{tabular}{c}\toprule
\begin{minipage}[c]{0.8\linewidth}
\insertminitoc\relax
\end{minipage}\\\bottomrule
\end{tabular}
\end{center}
```

where \minitocFmt is a command that expands to some formatting, see demo files.

¶ The mini-toc format environment: minitocfmt. To help facilitate designing and declaring the mini-toc format, we define the minitocfmt environment. The environment defines a command (⟨\cmd⟩) that contains all the formatting information for the mini-toc. The body of the environment consists of a series of \declaretocfmt{⟨*toc-fmt*⟩} declarations. Within argument of \declaretocfmt, \@W is an alias for \numBoxWidth and \@D is an alias for \sl@dots. If \@D appears

(`@D` = `@dottedtocline`, a dotted line is created in the usual LaTeX manner. `\@N` is an alias `\sl@sNumFmt` and `\@P` is an alias for `\sl@pNumFmt`. All are optional.

`\@A{`⟨*various*⟩`}` is a command that is not used very often, but is available when needed. The argument ⟨*various*⟩ is various commands to support the mintoc being generated.

pg num box width `\@PW{`⟨*length*⟩`}` Within the argument of `\@A`, insert `\@PW{`⟨*length*⟩`}` to set the width of the box that contains the page number (`\@pnumwidth`). The value set by LaTeX is `1.55em`.

dots separation `\@DS{`⟨*num*⟩`}` The ⟨*num*⟩ determines the separation between dots for a TOC entry that uses a dotted rule line. This command is only recognized within the argument of `\@A`. The default is 4.5.

right margin of title `\@R{`⟨*length*⟩`}` is a convenience command, it takes its argument and defines the LaTeX command `\@tocrmarg`, which sets the right margin for the sec-title. The length set by LaTeX is `2.55em`. The ⟨*length*⟩ of `\@R` should be *larger than* the ⟨*length*⟩ set by `\@PW`.

`\declaretocfmt{`⟨*sec-name*⟩`}{`⟨*various*⟩`}` formats all ⟨*sec-name*⟩ (section, subsection, etc.) entries.

A 'typical' table of contents entry has the form:

⟨*sec-num*⟩ ⟨*title-heading*⟩ . . . . ⟨*pg-num*⟩

Within the ⟨*various*⟩ argument, there are a number of commands that are recognized:

sec num box width `\@W{`⟨*length*⟩`}` is the width of the box that encloses ⟨*sec-num*⟩. Normally, all lengths are measured in `em` units (`\@W{`⟨*num*⟩`em}`). The default length is `2.5em`

use dots `\@D{`⟨*length*⟩`}` is the amount to indent prior to ⟨*sec-num*⟩. Again, `em` units preferred (`\@D{`⟨*num*⟩`em}`). When the `\@D` command is present in the argument, a dotted line is to be used for the entry (this is the norm). If `\@D` not present, there is an opportunity within the ⟨*various*⟩ argument to create a custom entry.

no dots `\@B{`⟨*length*⟩`}` Same as `\@D`, but no dotted leaders are created.

fmt sec num `\@N{`⟨*fmt*⟩`}` is the formatting for ⟨*sec-num*⟩. You can pass a command with one argument that will operate on the section number; for example, `\@N{\textbf}`, `\@N{\color{blue}}`, or `\@N{\color{blue}\textbf}`. Note that changing the style to bold might require a corresponding change in `\@W`.

fmt title `\@F{`⟨*fmt*⟩`}` is the formatting for the title heading of the current section; for example, `\@F{\bfseries}` turns all heading, for this ⟨*sec-name*⟩, bold.

fmt pg num `\@P{`⟨*fmt*⟩`}` is the formatting for the page number (⟨*pg-num*⟩). You can pass a command with one argument that will operate on the page number

When hyperref is loaded with the `colorlinks` option, we cannot change the color of the page number (see the discussion of `\@A` above), but `\@P{\textit}` changes the numbers to italics. If hyperref is not loaded, `\@P{\color{red}\textit}` changes page numbers to a red italic.

right margin of title `\@R{`⟨*length*⟩`}` is a convenience command, it takes its argument and defines the LaTeX command `\@tocrmarg`, which sets the right margin for the sec-title. The length set by LaTeX is `2.55em`. Setting `\@R` within the ⟨*various*⟩ argument of `\declaretocfmt` affects the current section level as well as all lower section levels. If you want to make this 'local' change, you need to put `\@R` back to its default of `2.55em` locally for other declarations.

TOC number `\@E` Within the `minitocfmt` environment, the command `\@E` expands to the current TOC entry number of the TOC entry being read in.

link anchor `\@L` This macro expands to the hyperref anchor of the page entry reference, it is empty if hyperref is not loaded.

pg number `\@Pg` This macro expands to the page number this entry references.

Usually, the ⟨*length*⟩ argument is measured in `em` units (⟨*num*⟩`em`).

```
\newtoks\mtoc@toks
\newtoks\mtocs@toks@
```

Within the `minitocfmt`, `\@D` is `\let` to `\sl@dots`.

```
\def\sl@dots#1{\def\sl@@dots{%
  \@dottedtocline{\sl@current}{#1}{\mtoc@numBoxWidth}}}
\let\sl@@dots\@gobble
\def\sl@nodots#1{\def\sl@@dots{%
  \no@dottedtocline{\sl@current}{#1}{\mtoc@numBoxWidth}}}
\let\sl@@nodots\@gobble
```

Within the `minitocfmt`, `\@F` is `\let` to *\@EntryFmt*.

```
\def\sl@EntryFmt#1{\def\sl@@EntryFmt{#1}} % dps
\let\sl@@EntryFmt\relax % dps
```

Within the `minitocfmt`, `\@N` is `\let` to `\sl@sNumFmt`.

```
\def\sl@sNumFmt#1{\def\sl@@sNumFmt{#1}}
\let\sl@@sNumFmt\relax
```

Within the `minitocfmt`, `\@P` is `\let` to `\sl@pNumFmt`.

```
\def\sl@pNumFmt#1{\def\sl@@pNumFmt{#1}}
\let\sl@@pNumFmt\relax
\def\sl@tocrmarg#1{\def\@tocrmarg{#1}}
\def\sl@dotsep#1{\def\@dotsep{#1}}
\def\mtoc@star#1*#2\@nil{\def\@rgii{#2}\ifx\@rgii\@empty
  \let\mtoc@@star\mtoc@Zero\else\let\mtoc@@star\mtoc@One\fi}
```

¶¶ The `\declaretocfmt` command defined. The `\declaretocfmt` is used to designed how a mini-toc entry is displayed.

`\declaretocfmt{`⟨*sec-name*⟩`}{`⟨*various*⟩`}` The aeb-mintoc way of declaring the formatting for a toc ⟨*sec-name*⟩ entry. The ⟨*various*⟩ argument consists of various combinations of `\@W`, `\@D`, `\@N`, and `\@P`.

```
\long\def\declaretocfmt#1#2{%
  \xdef\sl@current{\@nameuse{sl@#1}}%
  \global\@namedef{\mtoc@CmdName @l@#1}##1##2{%
    \normalfont\normalcolor\let\@E\TOCEntryNum
    \let\@L\mtoc@HY@anchor\let\@Pg\mtoc@PgNum
    \let\sl@@dots\@empty\let\sl@@sNumFmt\relax
    \let\sl@@pNumFmt\relax\let\sl@@EntryFmt\relax
    \let\@W\numBoxWidth\let\@R\sl@tocrmarg\let\@D\sl@dots
    \let\@B\sl@nodots\let\@F\sl@EntryFmt\let\@N\sl@sNumFmt
    \let\@P\sl@pNumFmt
    #2\ifx\sl@@dots\@empty\let\sl@next\relax\else
      \mtoc@star#1*\@nil % dps
      \ifx\mtoc@@star\mtoc@Zero
        \def\sl@next{\sl@@dots{##1}{\sl@@pNumFmt{##2}}}\else
        \def\sl@next{\sl@@dots{\sl@@EntryFmt##1}{\sl@@pNumFmt{##2}}}\fi
    \fi\sl@next}%
  \edef\x{\expandafter\noexpand\csname l@#1\endcsname}%
  \edef\y{\expandafter\noexpand\csname\mtoc@CmdName @l@#1\endcsname}%
  \edef\mtoc@@tmp{\the\mtoc@toks\let\expandafter\noexpand\x=
  \expandafter\noexpand\y}
  \global\mtoc@toks=\expandafter{\mtoc@@tmp}}
\def\mtoc@getCmdName#1{\edef\mtoc@CmdName{\expandafter
  \@gobble\string#1}}
```

¶¶ The `minitocfmt` environment defined. Is a 'simplified' way of designing toc entries.

`minitocfmt{\`⟨*cmdName*⟩`}` The definition of the environment. The argument is a command that will hold the expanded content of the environment. The body of the environment consists of one or more `\declaretocfmt` commands.

```
\newenvironment{minitocfmt}[1]{\makeatletter
  \gdef\@mtoc@cmd@@{#1}\let\@A\mtoc@addto
```

The `\mtoc@getCmdName` returns the `cmdName` (without backslash). `cmdName` is used the creating command sequences, using to this definition.

```
  \mtoc@getCmdName{#1}\mtoc@toks={\let\@PW\mtoc@PW\let\@DS\sl@dotsep
    \let\@R\sl@tocrmarg}%
```

The body of the environment consists of one or more `\declaretocfmt` commands, these commands contribute to `\mtoc@toks`. `\mtoc@toks` consists of all the formatting declarations requested.

```
}{\expandafter\xdef\@mtoc@cmd@@{\the\mtoc@toks}}
```

`\mtoc@addto` is a macro to add to the declarations. Within `minitocfmt` is `\@A` is `\let` to `\mtoc@addto`.

```
\def\mtoc@addto#1{\edef\mtoc@@tmp{\the\mtoc@toks}%
  \global\mtoc@toks=\expandafter{\mtoc@@tmp #1}}
```

```
\def\mtoc@PW#1{\def\@pnumwidth{#1}}
```

Here is code from `latex.ltx` for `\@dottedtocline`, we modify it so there are no leaders.

```
\def\no@dottedtocline#1#2#3#4#5{%
  \ifnum #1>\c@tocdepth \else
    \vskip \z@ \@plus.2\p@
    {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
     \parindent #2\relax\@afterindenttrue
     \interlinepenalty\@M
     \leavevmode
     \@tempdima #3\relax
     \advance\leftskip \@tempdima \null\nobreak\hskip -\leftskip
     {#4}\nobreak
%     \leaders\hbox{$\m@th
%        \mkern \@dotsep mu\hbox{.}\mkern \@dotsep
%       mu$}\hfill
% Insert an \hfill
     \hfill
     \nobreak
     \hb@xt@\@pnumwidth{\hfil\normalfont \normalcolor #5}%
     \par}%
  \fi}
```

⟨/package⟩

3 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.

4 Change History

v1.2 (2018/08/29)
General: Created `aeb-minitoc.ins` 2

v1.3 (2018/08/29)
General: Remove hyperref as a requirement 2

v1.4 (2018/08/29)
General: Some renaming of commands 2

v1.6 (2018/09/21)
General: Code cleanup in prepreparation for release 2

v1.7 (2018/09/29)
General: Delay redefinition of `\addtocontents` until beginning of document, to avoid incompatability with siunitx. 6