

# The `atenddvi` package

Heiko Oberdiek  
<heiko.oberdiek at googlemail.com>

2007/04/17 v1.1

## Abstract

L<sup>A</sup>T<sub>E</sub>X offers `\AtBeginDvi`. This package `atenddvi` provides the counterpart `\AtEndDvi`. The execution of its argument is delayed to the end of the document at the end of the last page. Thus `\special` and `\write` remain effective, because they are put into the last page. This is the main difference to `\AtEndDocument`.

## Contents

<b>1</b>	<b>Documentation</b>	<b>1</b>
<b>2</b>	<b>Implementation</b>	<b>2</b>
<b>3</b>	<b>Installation</b>	<b>3</b>
3.1	Download . . . . .	3
3.2	Bundle installation . . . . .	4
3.3	Package installation . . . . .	4
3.4	Refresh file name databases . . . . .	4
3.5	Some details for the interested . . . . .	4
<b>4</b>	<b>History</b>	<b>5</b>
	[2007/03/20 v1.0] . . . . .	5
	[2007/04/17 v1.1] . . . . .	5
<b>5</b>	<b>Index</b>	<b>5</b>

## 1 Documentation

`\AtEndDvi {⟨code⟩}`

Macro `\AtEndDvi` provides a hook mechanism to put `⟨code⟩` at the end of the last output page. It is the logical counterpart to `\AtBeginDvi`. Despite the name the output type DVI, PDF or whatever does not matter.

Unlike `\AtBeginDvi` the `⟨code⟩` is not put in a box and therefore executed immediately. The hook for `\AtEndDvi` is based on a macro similar to `\AtBeginDocument` or `\AtEndDocument`. The execution of `⟨code⟩` is delayed until the hook is executed on the last page.

Commands such as `\special` or `\write` (not the `\immediate` variant) must go as nodes into the contents of a page to have the desired effect. When the hook for `\AtEndDocument` is executed, the last intended page may already be shipped out. Therefore `\special` or `\write` cannot be used in a reliable way without generating new page.

This gap is closed by `\AtEndDvi` of this package `atenddvi`. If the document is compiled the first time, the package remembers the last page in a reference. In the

second run, it puts the hook on the page that has been detected in the previous run as last page. The package detects if the number of pages has changed, and then generates a warning to rerun L<sup>A</sup>T<sub>E</sub>X.

## 2 Implementation

```

1 <*package>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{atenddvi}%
4 [2007/04/17 v1.1 At end DVI hook (H0)]%

    Load the required packages
5 \RequirePackage{zref-abspage,zref-lastpage}[2007/03/19]
6 \RequirePackage{atbegshi}

\AtEndDvi@Hook Macro \AtEndDvi@Hook is the data storage macro for the code that is executed
later at end of the last page.
7 \let\AtEndDvi@Hook\@empty

\AtEndDvi Macro \AtEndDvi is called in the same way as \AtBeginDocument. The argument
is added to the hook macro.
8 \newcommand*{\AtEndDvi}{%
9   \g@addto@macro\AtEndDvi@Hook
10 }

\AtEndDvi@AtBeginShipout
11 \def\AtEndDvi@AtBeginShipout{%
12   \begingroup

    The reference 'LastPage' is marked used. If the reference is not yet defined, then
    the user gets the warning because of the undefined reference and the rerun warning
    at the end of the compile run. However, we do not need a warning each page, the
    first page is enough.
13   \ifnum\value{abspage}=1 %
14     \zref@refused{LastPage}%
15   \fi

    The current absolute page number is compared with the absolute page number of
    the reference 'LastPage'.
16   \ifnum\zref@extractdefault{LastPage}{abspage}{0}=\value{abspage}%

\AtEndDvi@LastPage We found the right page and remember it in a macro.
17   \xdef\AtEndDvi@LastPage{\number\value{abspage}}%

    The hook of \AtEndDvi is now put on the last page after the contents of the page.
18   \global\setbox\AtBeginShipoutBox=\vbox{%
19     \hbox{%
20       \box\AtBeginShipoutBox
21       \setbox\AtBeginShipoutBox=\hbox{%
22         \begingroup
23         \AtEndDvi@Hook
24         \endgroup
25       }%
26       \wd\AtBeginShipoutBox=\z@
27       \ht\AtBeginShipoutBox=\z@
28       \dp\AtBeginShipoutBox=\z@
29       \box\AtBeginShipoutBox
30     }%
31   }%

    We do not need the every page hook.
32   \global\let\AtEndDvi@AtBeginShipout\@empty

```

```

33      \global\let\AtEndDvi\@gobble
Make a protocol entry, which page is used by this package as last page.
34      \let\on@line\@empty
35      \PackageInfo{atendddvi}{Last page = \AtEndDvi@LastPage}%
36      \fi
37  \endgroup
38 }

```

```

39 \def\AtEndDvi@AtBeginDocument{%
40   \AtBeginShipout{\AtEndDvi@AtBeginShipout}%

```

During the first reading of the `.aux` file, `\AtEndDvi@Check` is disabled, its real meaning is assigned afterwards.

```

41 \if@filesw
42   \immediate\write\@mainaux{%
43     \string\providecommand\string\AtEndDvi@Check{}}%
44   }%
45   \immediate\write\@mainaux{%
46     \string\AtEndDvi@Check
47   }%
48 \fi
49 \let\AtEndDvi@Check\AtEndDvi@CheckImpl
50 }

```

```
51 \AtBeginDocument{\AtEndDvi@AtBeginDocument}
```

```

52 \def\AtEndDvi@CheckImpl{%
53   \@ifundefined{AtEndDvi@LastPage}{%
54     \PackageWarningNoLine{atendddvi}{%
55       Rerun LaTeX, last page not yet found%
56     }%
57   }{%
58     \ifnum\AtEndDvi@LastPage=\value{abspage}%
59     \else
60       \PackageWarningNoLine{atendddvi}{%
61         Rerun LaTeX, last page has changed%
62       }%
63     \fi
64   }%
65 }

```

66 `</package>`

**Bundle.** All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](#)

*TDS* refers to the standard “A Directory Structure for  $\text{\TeX}$  Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

## 3.2 Bundle installation

**Unpacking.** Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

**Script installation.** Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

## 3.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting `docstrip` archive. The files are extracted by running the `.dtx` through plain- $\text{\TeX}$ :

```
tex atenddvi.dtx
```

**TDS.** Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

```
atenddvi.sty → tex/latex/oberdiek/atenddvi.sty
atenddvi.pdf → doc/latex/oberdiek/atenddvi.pdf
atenddvi.dtx → source/latex/oberdiek/atenddvi.dtx
```

If you have a `docstrip.cfg` that configures and enables `docstrip`’s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

## 3.4 Refresh file name databases

If your  $\text{\TeX}$  distribution (te $\text{\TeX}$ , mi $\text{\TeX}$ , ...) relies on file name databases, you must refresh these. For example, te $\text{\TeX}$  users run `texhash` or `mktextlsr`.

## 3.5 Some details for the interested

**Attached source.** The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk atenddvi.pdf unpack_files output .
```

**Unpacking with L<sup>A</sup>T<sub>E</sub>X.** The .dtx chooses its action depending on the format:

**plain-T<sub>E</sub>X:** Run docstrip and extract the files.

**L<sup>A</sup>T<sub>E</sub>X:** Generate the documentation.

If you insist on using L<sup>A</sup>T<sub>E</sub>X for docstrip (really, docstrip does not need L<sup>A</sup>T<sub>E</sub>X), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{atenddvi.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

**Generating the documentation.** You can use both the .dtx or the .drv to generate the documentation. The process can be configured by the configuration file ltxdoc.cfg. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfL<sup>A</sup>T<sub>E</sub>X:

```
pdflatex atenddvi.dtx
makeindex -s gind.ist atenddvi.idx
pdflatex atenddvi.dtx
makeindex -s gind.ist atenddvi.idx
pdflatex atenddvi.dtx
```

## 4 History

[2007/03/20 v1.0]

- First version.

[2007/04/17 v1.1]

- Package atbegshi replaces package everyshi.

## 5 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.

Symbols		
\@empty	7, 32, 34	\AtEndDvi@LastPage . . . . . 17, 35, 58
\@gobble	33	<b>B</b>
\@ifundefined	53	\box . . . . . 20, 29
\@mainaux	42, 45	<b>D</b>
<b>A</b>		\dp . . . . . 28
\AtBeginDocument	51	<b>G</b>
\AtBeginShipout	40	\g@addto@macro . . . . . 9
\AtBeginShipoutBox		<b>H</b>
	18, 20, 21, 26, 27, 28, 29	\hbox . . . . . 19, 21
\AtEndDvi	1, 8, 33	\ht . . . . . 27
\AtEndDvi@AtBeginDocument	39	<b>I</b>
\AtEndDvi@AtBeginShipout	11, 40	\if@filesw . . . . . 41
\AtEndDvi@Check	41	\ifnum . . . . . 13, 16, 58
\AtEndDvi@CheckImpl	49, 52	\immediate . . . . . 42, 45
\AtEndDvi@Hook	7, 9, 23	

<b>N</b>		<b>S</b>	
\NeedsTeXFormat .....	2	\setbox .....	18, 21
\newcommand .....	8	<b>V</b>	
\number .....	17	\value .....	13, 16, 17, 58
<b>O</b>		\vbox .....	18
\on@line .....	34	<b>W</b>	
<b>P</b>		\wd .....	26
\PackageInfo .....	35	\write .....	42, 45
\PackageWarningNoLine .....	54, 60	<b>Z</b>	
\providecommand .....	43	\z@ .....	26, 27, 28
\ProvidesPackage .....	3	\zref@extractdefault .....	16
<b>R</b>		\zref@refused .....	14
\RequirePackage .....	5, 6		