

# The `catchfile` package

Heiko Oberdiek  
<oberdiek@uni-freiburg.de>

2007/05/30 v1.0

## Abstract

This package catches the contents of a file and puts it in a macro. It requires  $\varepsilon$ -TeX. Both L<sup>A</sup>T<sub>E</sub>X and plain-T<sub>E</sub>X are supported.

## Contents

<b>1</b>	<b>Documentation</b>	<b>1</b>
<b>2</b>	<b>Implementation</b>	<b>2</b>
2.1	Reload check and package identification . . . . .	2
2.2	Catcodes . . . . .	3
2.3	Preparations . . . . .	3
2.4	Looking for primitive <code>\input</code> . . . . .	3
2.5	Input file check . . . . .	4
2.6	Catch file contents . . . . .	5
<b>3</b>	<b>Test</b>	<b>5</b>
3.1	L <sup>A</sup> T <sub>E</sub> X . . . . .	5
3.2	plain-T <sub>E</sub> X . . . . .	6
<b>4</b>	<b>Installation</b>	<b>6</b>
4.1	Download . . . . .	6
4.2	Bundle installation . . . . .	7
4.3	Package installation . . . . .	7
4.4	Refresh file name databases . . . . .	7
4.5	Some details for the interested . . . . .	7
<b>5</b>	<b>History</b>	<b>8</b>
	[2007/05/30 v1.0] . . . . .	8
<b>6</b>	<b>Index</b>	<b>8</b>

## 1 Documentation

The package relies on  $\varepsilon$ -TeX's `\everyeof`. Otherwise it aborts with an error message.

<code>\CatchFileDef</code> $\{\langle cmd \rangle\}$ $\{\langle file name \rangle\}$ $\{\langle setup \rangle\}$ <code>\CatchFileEdef</code> $\{\langle cmd \rangle\}$ $\{\langle file name \rangle\}$ $\{\langle setup \rangle\}$
---

Macro  $\langle cmd \rangle$  is defined with the contents of file  $\langle file name \rangle$ . `\CatchFileDef` uses `\def`, `\CatchFileEdef` `\edef` for the definition. Additional setup code for setting catcodes or treatment of line ends can be given in code  $\langle setup \rangle$ . See the test files for an example.

## 2 Implementation

```
1 (*package)
```

### 2.1 Reload check and package identification

Reload check, especially if the package is not used with L<sup>A</sup>T<sub>E</sub>X.

```
2 \begingroup
3   \catcode44 12 % ,
4   \catcode45 12 % -
5   \catcode46 12 % .
6   \catcode58 12 % :
7   \catcode64 11 % @
8   \expandafter\let\expandafter\x\csname ver@catchfile.sty\endcsname
9   \ifcase 0%
10    \ifx\x\relax % plain
11    \else
12      \ifx\x\empty % LaTeX
13      \else
14        1%
15      \fi
16    \fi
17  \else
18    \expandafter\ifx\csname PackageInfo\endcsname\relax
19      \def\x#1#2{%
20        \immediate\write-1{Package #1 Info: #2.}%
21      }%
22    \else
23      \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
24    \fi
25    \x{catchfile}{The package is already loaded}%
26  \endgroup
27  \expandafter\endinput
28  \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31   \catcode40 12 % (
32   \catcode41 12 % )
33   \catcode44 12 % ,
34   \catcode45 12 % -
35   \catcode46 12 % .
36   \catcode47 12 % /
37   \catcode58 12 % :
38   \catcode64 11 % @
39   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
40     \def\x#1#2#3[#4]{\endgroup
41       \immediate\write-1{Package: #3 #4}%
42       \xdef#1{#4}%
43     }%
44   \else
45     \def\x#1#2[#3]{\endgroup
46       #2[#{#3}]%
47       \ifx#1\relax
48         \xdef#1{#3}%
49       \fi
50     }%
51   \fi
52   \expandafter\x\csname ver@catchfile.sty\endcsname
53   \ProvidesPackage{catchfile}%
54   [2007/05/30 v1.0 Catches the contents of a file (H0)]
```

## 2.2 Catcodes

```

55 \expandafter\edef\csname CatchFile@End\endcsname{%
56   \catcode44 \the\catcode44\relax % ,
57   \catcode45 \the\catcode45\relax % -
58   \catcode46 \the\catcode46\relax % .
59   \catcode61 \the\catcode61\relax % =
60   \catcode64 \the\catcode64\relax % @
61   \catcode96 \the\catcode96\relax % ‘
62   \noexpand\endinput
63 }
64 \catcode44 12 % ,
65 \catcode45 12 % -
66 \catcode46 12 % .
67 \catcode61 12 % =
68 \catcode64 11 % @
69 \catcode96 12 % ‘

```

## 2.3 Preparations

```

70 \begingroup\expandafter\expandafter\expandafter\endgroup
71 \expandafter\ifx\csname RequirePackage\endcsname\relax
72   \input infwarerr.sty\relax
73 \else
74   \RequirePackage{infwarerr}%
75 \fi

```

Check for  $\varepsilon$ -TeX's `\everyeof`.

```

76 \begingroup
77   \escapechar=92\relax
78   \edef\TestString{\string\everyeof}%
79   \edef\TestMeaning{\meaning\everyeof}%
80   \ifx\TestString\TestMeaning
81     \else
82       \@PackageErrorNoLine{catchfile}{%
83         Cannot find e-TeX's \string\everyeof, \MessageBreak
84         package loading is aborted%
85       }\@ehd
86     \endgroup
87     \expandafter\CatchFile@End
88   \fi
89 \endgroup

```

## 2.4 Looking for primitive `\input`

`\CatchFile@Input` The package needs the expandable primitive `\input`. However there are formats that redefine it. For example, L<sup>A</sup>T<sub>E</sub>X's `\input` is not expandable, but it stores the primitive in `\@@input`. The third possibility is `\pdfprimitive`, introduced in pdf<sub>T</sub>E<sub>X</sub> 1.40.0.

Thus we try to find the primitive and store it in `\CatchFile@Input`. If it is used, it must be expanded twice (because of the solution with `\pdfprimitive`).

```

90 \begingroup
91   \def\Check#1#2#3#4\endgroup{%
92     \edef\TestString{\string#1}%
93     \edef\TestMeaning{\meaning#2}%
94     \ifx\TestString\TestMeaning
95       \endgroup
96       \let\CatchFile@Primitive#2%
97       \def\CatchFile@Input{\CatchFile@Primitive#3}%
98     \else
99       #4\endgroup
100   \fi
101 }%
102 \Check\input\input{%

```

```

103 \Check\input\@input{}%
104 \Check\pdfprimitive\pdfprimitive\input
105 \@PackageErrorNoLine{%
106   Cannot find primitive \string\input,\MessageBreak
107   package loading is aborted%
108 }\@ehd
109 \csname endgroup\endcsname
110 \CatchFile@End
111 \endgroup

```

## 2.5 Input file check

\CatchFile@CheckFileExists

```

112 \begingroup\expandafter\expandafter\expandafter\endgroup
113 \expandafter\ifx\csname IfFileExists\endcsname\relax
114 \begingroup\expandafter\expandafter\expandafter\endgroup
115 \expandafter\ifx\csname pdffilesize\endcsname\relax
116   \def\CatchFile@CheckFileExists#1{%
117     \expandafter\ifx\csname @inputcheck\endcsname\relax
118       \csname newread\endcsname\@inputcheck
119     \fi
120     \openin\@inputcheck#1\relax
121     \ifeof\@inputcheck
122       \let\CatchFile@File\relax
123     \else
124       \closein\@inputcheck
125       \def\CatchFile@File{#1}%
126     \fi
127   }%
128 \else
129   \def\CatchFile@CheckFileExists#1{%
130     \expandafter\ifx\expandafter\\pdffilesize{#1}\\%
131       \let\CatchFile@File\relax
132     \else
133       \def\CatchFile@File{#1}%
134     \fi
135   }%
136 \fi
137 \else
138   \def\CatchFile@CheckFileExists#1{%
139     \IfFileExists{#1}{%
140       \expandafter\CatchFile@DefFile\@filef@und\@nil
141       \begingroup\expandafter\expandafter\expandafter\endgroup
142       \expandafter\ifx\csname @addtofilelist\endcsname\relax
143       \else
144         \@addtofilelist\CatchFile@File
145       \fi
146     }{%
147       \let\CatchFile@File\relax
148     }%
149   }%
150   \def\CatchFile@DefFile#1 \@nil{%
151     \def\CatchFile@File{#1}%
152   }%
153 \fi

```

\CatchFileNotFound

```

154 \def\CatchFile@NotFound#1#2{%
155   \def#1{%
156     \@PackageError{catchfile}{%
157       File ‘#2’ not found%
158     }\@ehc

```

159 }

## 2.6 Catch file contents

\CatchFileEdef

```

160 \long\def\CatchFileEdef#1#2#3{%
161   \CatchFile@CheckFileExists{#2}%
162   \ifx\CatchFile@File\relax
163     \CatchFile@NotFound{#1}{#2}%
164   \else
165     \begingroup
166     \everyeof{\noexpand}%
167     #3%
168     \xdef\CatchFile@Contents{\CatchFile@Input\CatchFile@File\space}%
169   \endgroup
170   \let#1\CatchFile@Contents
171 \fi
172 }
```

\CatchFileDef

```

173 \long\def\CatchFileDef#1#2#3{%
174   \CatchFile@CheckFileExists{#2}%
175   \ifx\CatchFile@File\relax
176     \CatchFile@NotFound{#1}{#2}%
177   \else
178     \begingroup
179     \everyeof\expandafter{%
180       \CatchFile@EOF
181       \noexpand
182     }%
183     \expandafter\long\expandafter\def\expandafter\CatchFile@Do
184       \expandafter##\expandafter1\CatchFile@EOF{%
185       \endgroup
186       \def#1{##1}%
187     }%
188     #3%
189     \expandafter\expandafter\expandafter\CatchFile@Do
190     \CatchFile@Input\CatchFile@File\relax
191   \fi
192 }
```

\CatchFile@EOF If the file is read the catcode mappings are fixed. This means that the same character cannot occur inside the file with different catcodes. Thus we use as end of file marker the at sign twice with different catcodes.

```

193 \begingroup
194 \lccode65=64 % lowercase('A') = '@'
195 \lccode66=64 % lowercase('B') = '@'
196 \catcode65=8 % catcode('A') = subscript
197 \catcode66=3 % catcode('B') = math shift
198 \lowercase{\endgroup
199 \def\CatchFile@EOF{AB}%
200 }
```

```

201 \CatchFile@End
202 \endpackage
```

## 3 Test

### 3.1 L<sup>A</sup>T<sub>E</sub>X

203 \test1

```

204 \NeedsTeXFormat{LaTeX2e}
205 \nofiles
206 \listfiles
207 \documentclass{minimal}
208 \usepackage{catchfile}[2007/05/30]
209 \makeatletter
210 \def\mysetup{%
211   \let\do\@makeoother
212   \dospecials
213 }
214 \makeatother
215 \begin{document}
216
217 \CatchFileDef\contents{catchfile.sty}\mysetup
218 \typeout{\meaning\contents}
219
220 \CatchFileEdef\contents{catchfile.sty}{%
221   \mysetup
222   \def\par{^^J}%
223   \obeylines
224 }
225 \typeout{\contents}
226 \end{document}
227 </test1>

```

## 3.2 plain-TeX

```

228 <*test2>
229 \def\msg#{\immediate\write16}
230 \newlinechar=10 %
231 \input catchfile.sty\relax
232
233 \def\mysetup{%
234   \def\do##1{%
235     \catcode'\##1=12\relax
236   }%
237   \dospecials
238 }
239
240 \CatchFileDef\contents{catchfile.sty}\mysetup
241 \msg{\meaning\contents}
242
243 \CatchFileEdef\contents{catchfile.sty}{%
244   \mysetup
245   \def\par{^^J}%
246   \obeylines
247 }
248 \msg{\contents}
249
250 \csname @@end\endcsname
251 \end
252 </test2>

```

## 4 Installation

### 4.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

[CTAN:macros/latex/contrib/oberdiek/catchfile.dtx](http://ftp.ctan.org/tex-archive/macros/latex/contrib/oberdiek/catchfile.dtx) The source file.

[CTAN:macros/latex/contrib/oberdiek/catchfile.pdf](http://ftp.ctan.org/tex-archive/macros/latex/contrib/oberdiek/catchfile.pdf) Documentation.

---

<sup>1</sup><http://ftp.ctan.org/tex-archive/>

**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:macros/latex/contrib/oberdiek/oberdiek-tds.zip

## 4.2 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/
```

## 4.3 Package installation

**Unpacking.** The .dtx file is a self-extracting docstrip archive. The files are extracted by running the .dtx through plain-TeX:

```
tex catchfile.dtx
```

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

```
catchfile.sty      → tex/generic/oberdiek/catchfile.sty
catchfile.pdf      → doc/latex/oberdiek/catchfile.pdf
catchfile-test1.tex → doc/latex/oberdiek/catchfile-test1.tex
catchfile-test2.tex → doc/latex/oberdiek/catchfile-test2.tex
catchfile.dtx      → source/latex/oberdiek/catchfile.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.

## 4.4 Refresh file name databases

If your TeX distribution (teTeX, mikTeX, ...) relies on file name databases, you must refresh these. For example, teTeX users run texhash or mktexlsr.

## 4.5 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 catchfile.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-TeX:** 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{catchfile.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 catchfile.dtx
makeindex -s gind.ist catchfile.idx
pdflatex catchfile.dtx
makeindex -s gind.ist catchfile.idx
pdflatex catchfile.dtx
```

## 5 History

[2007/05/30 v1.0]

- First version.

## 6 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	
<code>\@@input</code> .....	103
<code>\@PackageError</code> .....	156
<code>\@PackageErrorNoLine</code> .....	82, 105
<code>\@addtofilelist</code> .....	144
<code>\@ehc</code> .....	158
<code>\@ehd</code> .....	85, 108
<code>\@filef@und</code> .....	140
<code>\@inputcheck</code> .....	118, 120, 121, 124
<code>\@makeoother</code> .....	211
<code>\@nil</code> .....	140, 150
<code>\@</code> .....	130
B	
<code>\begin</code> .....	215
C	
<code>\CatchFile@CheckFileExists</code> .....	112, 161, 174
<code>\CatchFile@Contents</code> .....	168, 170
<code>\CatchFile@DefFile</code> .....	140, 150
<code>\CatchFile@Do</code> .....	183, 189
<code>\CatchFile@End</code> .....	87, 110, 201
<code>\CatchFile@EOF</code> .....	180, 184, <u>193</u>
<code>\CatchFile@File</code> .....	122, 125, 131, 133, 144, 147, 151, 162, 168, 175, 190
<code>\CatchFile@Input</code> .....	90, 168, 190
<code>\CatchFile@NotFound</code> ...	154, 163, 176
<code>\CatchFile@Primitive</code> .....	96, 97
<code>\CatchFileDef</code> .....	1, <u>173</u> , 217, 240
<code>\CatchFileEdef</code> .....	<u>160</u> , 220, 243
<code>\CatchFileNotFound</code> .....	<u>154</u>
<code>\catcode</code> .....	3, 4, 5, 6, 7, 31, 32, 33, 34, 35, 36,
	37, 38, 56, 57, 58, 59, 60, 61, 64, 65, 66, 67, 68, 69, 196, 197, 235
<code>\Check</code> .....	91, 102, 103, 104
<code>\closein</code> .....	124
<code>\contents</code> .....	217, 218, 220, 225, 240, 241, 243, 248
<code>\csname</code> .....	8, 18, 39, 52, 55, 71, 109, 113, 115, 117, 118, 142, 250
D	
<code>\do</code> .....	211, 234
<code>\documentclass</code> .....	207
<code>\dospecials</code> .....	212, 237
E	
<code>\empty</code> .....	12
<code>\end</code> .....	226, 251
<code>\endcsname</code> .....	8, 18, 39, 52, 55, 71, 109, 113, 115, 117, 118, 142, 250
<code>\endinput</code> .....	27, 62
<code>\escapechar</code> .....	77
<code>\everyeof</code> .....	78, 79, 83, 166, 179
I	
<code>\ifcase</code> .....	9
<code>\ifeof</code> .....	121
<code>\IfFileExists</code> .....	139
<code>\ifx</code> .....	10, 12, 18, 39, 47, 71, 80, 94, 113, 115, 117, 130, 142, 162, 175
<code>\immediate</code> .....	20, 41, 229
<code>\input</code> .....	72, 102, 103, 104, 106, 231
L	
<code>\lccode</code> .....	194, 195



\listfiles .....	206	\pdfprimitive .....	104
\lowercase .....	198	\ProvidesPackage .....	53
<b>M</b>			
\makeatletter .....	209	<b>R</b>	
\makeatother .....	214	\RequirePackage .....	74
\meaning .....	79, 93, 218, 241	<b>S</b>	
\MessageBreak .....	83, 106	\space .....	168
\msg .....	229, 241, 248	<b>T</b>	
\mysetup ..	210, 217, 221, 233, 240, 244	\TestMeaning .....	79, 80, 93, 94
<b>N</b>		\TestString .....	78, 80, 92, 94
\NeedsTeXFormat .....	204	\the .....	56, 57, 58, 59, 60, 61
\newlinechar .....	230	\typeout .....	218, 225
\nofiles .....	205	<b>U</b>	
<b>O</b>		\usepackage .....	208
\obeylines .....	223, 246	<b>W</b>	
\openin .....	120	\write .....	20, 41, 229
<b>P</b>		<b>X</b>	
\PackageInfo .....	23	\x .....	8, 10, 12, 19, 23, 25, 40, 45, 52
\par .....	222, 245		
\pdffilesize .....	130		