The **catchfile** package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/11/11 v1.2

Abstract

This package catches the contents of a file and puts it in a macro. It requires ε-TeX. Both LaTeX and plain-TeX are supported.

Contents

1 Documentation

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

`\CatchFileDef` {⟨*cmd*⟩} {⟨*file name*⟩} {⟨*setup*⟩}
`\CatchFileEdef` {⟨*cmd*⟩} {⟨*file name*⟩} {⟨*setup*⟩}

Macro ⟨*cmd*⟩ is defined with the contents of file ⟨*file name*⟩. `\CatchFileDef` uses `\def`, `\CatchFileEdef` `\edef` for the definition. Additional setup code for setting

catcodes or treatment of line ends can be given in code ⟨*setup*⟩. 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 LaTeX.

```
2 \begingroup
3   \catcode44 12 % ,
4   \catcode45 12 % -
5   \catcode46 12 % .
6   \catcode58 12 % :
7   \catcode64 11 % @
8   \catcode123 1 % {
9   \catcode125 2 % }
10  \expandafter\let\expandafter\x\csname ver@catchfile.sty\endcsname
11  \ifx\x\relax % plain-TeX, first loading
12  \else
13    \def\empty{}%
14    \ifx\x\empty % LaTeX, first loading,
15      % variable is initialized, but \ProvidesPackage not yet seen
16    \else
17      \catcode35 6 % #
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      \aftergroup\endinput
27    \fi
28  \fi
29 \endgroup
```

Package identification:

```
30 \begingroup
31   \catcode35 6 % #
32   \catcode40 12 % (
33   \catcode41 12 % )
34   \catcode44 12 % ,
35   \catcode45 12 % -
36   \catcode46 12 % .
37   \catcode47 12 % /
38   \catcode58 12 % :
39   \catcode64 11 % @
40   \catcode91 12 % [
41   \catcode93 12 % ]
42   \catcode123 1 % {
43   \catcode125 2 % }
44   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
45     \def\x#1#2#3[#4]{\endgroup
46       \immediate\write-1{Package: #3 #4}%
47       \xdef#1{#4}%
48     }%
49   \else
50     \def\x#1#2[#3]{\endgroup
51       #2[{#3}]%
```

```
52    \ifx#1\@undefined
53      \xdef#1{#3}%
54    \fi
55    \ifx#1\relax
56      \xdef#1{#3}%
57    \fi
58  }%
59  \fi
60 \expandafter\x\csname ver@catchfile.sty\endcsname
61 \ProvidesPackage{catchfile}%
62  [2007/11/11 v1.2 Catches the contents of a file (HO)]
```

2.2 Catcodes

```
63 \begingroup
64   \catcode123 1 % {
65   \catcode125 2 % }
66   \def\x{\endgroup
67     \expandafter\edef\csname CatchFile@AtEnd\endcsname{%
68       \catcode35 \the\catcode35\relax
69       \catcode64 \the\catcode64\relax
70       \catcode123 \the\catcode123\relax
71       \catcode125 \the\catcode125\relax
72     }%
73   }%
74 \x
75 \catcode35 6 % #
76 \catcode64 11 % @
77 \catcode123 1 % {
78 \catcode125 2 % }
79 \def\TMP@EnsureCode#1#2{%
80   \edef\CatchFile@AtEnd{%
81     \CatchFile@AtEnd
82     \catcode#1 \the\catcode#1\relax
83   }%
84   \catcode#1 #2\relax
85 }
86 \TMP@EnsureCode{39}{12}% '
87 \TMP@EnsureCode{44}{12}% ,
88 \TMP@EnsureCode{45}{12}% -
89 \TMP@EnsureCode{46}{12}% .
90 \TMP@EnsureCode{47}{12}% /
91 \TMP@EnsureCode{61}{12}% =
92 \TMP@EnsureCode{96}{12}% `
```

2.3 Preparations

```
93 \begingroup\expandafter\expandafter\expandafter\endgroup
94 \expandafter\ifx\csname RequirePackage\endcsname\relax
95   \input infwarerr.sty\relax
96 \else
97   \RequirePackage{infwarerr}[2007/09/09]%
98 \fi
```

Check for ε-TeX's \everyeof.

```
99 \begingroup
100   \escapechar=92\relax
101   \edef\TestString{\string\everyeof}%
102   \edef\TestMeaning{\meaning\everyeof}%
103   \ifx\TestString\TestMeaning
104   \else
105     \@PackageErrorNoLine{catchfile}{%
106       Cannot find e-TeX's \string\everyeof,\MessageBreak
107       package loading is aborted%
```

```
108       }\@ehd
109     \endgroup
110     \CatchFile@AtEnd
111     \expandafter\endinput
112   \fi
113 \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, LATEX's `\input` is not expandable, but it stores the primitive in `\@@input`. The third possibility is `\pdfprimitive`, introduced in pdfTEX 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`).

```
114 \begingroup
115   \def\Check#1#2#3#4\endgroup{%
116     \edef\TestString{\string#1}%
117     \edef\TestMeaning{\meaning#2}%
118     \ifx\TestString\TestMeaning
119       \endgroup
120       \let\CatchFile@Primitive#2%
121       \def\CatchFile@Input{\CatchFile@Primitive#3}%
122     \else
123       #4\endgroup
124     \fi
125   }%
126   \Check\input\input{}%
127   \Check\input\@@input{}%
128   \Check\pdfprimitive\pdfprimitive\input
129   \@PackageErrorNoLine{%
130     Cannot find primitive \string\input,\MessageBreak
131     package loading is aborted%
132   }\@ehd
133   \csname endgroup\endcsname
134   \CatchFile@AtEnd
135   \endinput
136 \endgroup
```

2.5 Input file check

\CatchFile@CheckFileExists

```
137 \begingroup\expandafter\expandafter\expandafter\endgroup
138 \expandafter\ifx\csname IfFileExists\endcsname\relax
139   \input pdftexcmds.sty\relax
140   \begingroup\expandafter\expandafter\expandafter\endgroup
141   \expandafter\ifx\csname pdf@filesize\endcsname\relax
142     \def\CatchFile@CheckFileExists#1{%
143       \expandafter\ifx\csname @inputcheck\endcsname\relax
144         \csname newread\endcsname\@inputcheck
145       \fi
146       \openin\@inputcheck#1\relax
147       \ifeof\@inputcheck
148         \let\CatchFile@File\relax
149       \else
150         \closein\@inputcheck
151         \def\CatchFile@File{#1}%
152       \fi
153     }%
154   \else
155     \def\CatchFile@CheckFileExists#1{%
156       \expandafter\expandafter\expandafter\ifx
```

4

```
157        \expandafter\expandafter\expandafter\relax\pdf@filesize{#1}\relax
158          \let\CatchFile@File\relax
159        \else
160          \def\CatchFile@File{#1}%
161        \fi
162      }%
163    \fi
164  \else
165    \def\CatchFile@CheckFileExists#1{%
166      \IfFileExists{#1}{%
167        \expandafter\CatchFile@DefFile\@filef@und\@nil
168        \begingroup\expandafter\expandafter\expandafter\endgroup
169        \expandafter\ifx\csname @addtofilelist\endcsname\relax
170        \else
171          \@addtofilelist\CatchFile@File
172        \fi
173      }{%
174        \let\CatchFile@File\relax
175      }%
176    }%
177    \def\CatchFile@DefFile#1 \@nil{%
178      \def\CatchFile@File{#1}%
179    }%
180  \fi
```

\CatchFileNotFound

```
181 \def\CatchFile@NotFound#1#2{%
182   \def#1{}%
183   \@PackageError{catchfile}{%
184     File '#2' not found%
185   }\@ehc
186 }
```

2.6   Catch file contents

\CatchFileEdef

```
187 \long\def\CatchFileEdef#1#2#3{%
188   \CatchFile@CheckFileExists{#2}%
189   \ifx\CatchFile@File\relax
190     \CatchFile@NotFound{#1}{#2}%
191   \else
192     \begingroup
193       \everyeof{\noexpand}%
194       #3%
195       \xdef\CatchFile@Contents{\CatchFile@Input\CatchFile@File\space}%
196     \endgroup
197     \let#1\CatchFile@Contents
198   \fi
199 }
```

\CatchFileDef

```
200 \long\def\CatchFileDef#1#2#3{%
201   \CatchFile@CheckFileExists{#2}%
202   \ifx\CatchFile@File\relax
203     \CatchFile@NotFound{#1}{#2}%
204   \else
205     \begingroup
206       \everyeof\expandafter{%
207         \CatchFile@EOF
208         \noexpand
209       }%
210       \expandafter\long\expandafter\def\expandafter\CatchFile@Do
```

```
211          \expandafter##\expandafter1\CatchFile@EOF{%
212        \endgroup
213        \def#1{##1}%
214      }%
215      #3%
216    \expandafter\expandafter\expandafter\CatchFile@Do
217    \CatchFile@Input\CatchFile@File\relax
218  \fi
219 }
```

\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.

```
220 \begingroup
221   \lccode65=64 % lowercase('A') = '@'
222   \lccode66=64 % lowercase('B') = '@'
223   \catcode65=8 % catcode('A') = subscript
224   \catcode66=3 % catcode('B') = math shift
225 \lowercase{\endgroup
226   \def\CatchFile@EOF{AB}%
227 }

228 \CatchFile@AtEnd
229 ⟨/package⟩
```

3   Test

3.1   Catcode checks for loading

```
230 ⟨∗test1⟩

231 \catcode`\{=1 %
232 \catcode`\}=2 %
233 \catcode`\#=6 %
234 \catcode`\@=11 %
235 \expandafter\ifx\csname count@\endcsname\relax
236   \countdef\count@=255 %
237 \fi
238 \expandafter\ifx\csname @gobble\endcsname\relax
239   \long\def\@gobble#1{}%
240 \fi
241 \expandafter\ifx\csname @firstofone\endcsname\relax
242   \long\def\@firstofone#1{#1}%
243 \fi
244 \expandafter\ifx\csname loop\endcsname\relax
245   \expandafter\@firstofone
246 \else
247   \expandafter\@gobble
248 \fi
249 {%
250   \def\loop#1\repeat{%
251     \def\body{#1}%
252     \iterate
253   }%
254   \def\iterate{%
255     \body
256       \let\next\iterate
257     \else
258       \let\next\relax
259     \fi
260     \next
261   }%
```

```
262  \let\repeat=\fi
263  }%
264  \def\RestoreCatcodes{}
265  \count@=0 %
266  \loop
267    \edef\RestoreCatcodes{%
268      \RestoreCatcodes
269      \catcode\the\count@=\the\catcode\count@\relax
270    }%
271  \ifnum\count@<255 %
272    \advance\count@ 1 %
273  \repeat
274
275  \def\RangeCatcodeInvalid#1#2{%
276    \count@=#1\relax
277    \loop
278      \catcode\count@=15 %
279    \ifnum\count@<#2\relax
280      \advance\count@ 1 %
281    \repeat
282  }
283  \expandafter\ifx\csname LoadCommand\endcsname\relax
284    \def\LoadCommand{\input catchfile.sty\relax}%
285  \fi
286  \def\Test{%
287    \RangeCatcodeInvalid{0}{47}%
288    \RangeCatcodeInvalid{58}{64}%
289    \RangeCatcodeInvalid{91}{96}%
290    \RangeCatcodeInvalid{123}{255}%
291    \catcode`\@=12 %
292    \catcode`\\=0 %
293    \catcode`\{=1 %
294    \catcode`\}=2 %
295    \catcode`\#=6 %
296    \catcode`\[=12 %
297    \catcode`\]=12 %
298    \catcode`\%=14 %
299    \catcode`\ =10 %
300    \catcode13=5 %
301    \LoadCommand
302    \RestoreCatcodes
303  }
304  \Test
305  \csname @@end\endcsname
306  \end
307  ⟨/test1⟩
```

3.2 LaTeX

```
308  ⟨∗test2⟩
309  \NeedsTeXFormat{LaTeX2e}
310  \nofiles
311  \listfiles
312  \documentclass{minimal}
313  \usepackage{catchfile}[2007/11/11]
314  \makeatletter
315  \def\mysetup{%
316    \let\do\@makeother
317    \dospecials
318  }
319  \makeatother
320  \begin{document}
321
```

```
322 \CatchFileDef\contents{catchfile.sty}\mysetup
323 \typeout{\meaning\contents}
324
325 \CatchFileEdef\contents{catchfile.sty}{%
326   \mysetup
327   \def\par{^^J}%
328   \obeylines
329 }
330 \typeout{\contents}
331 \end{document}
332 ⟨/test2⟩
```

3.3   plain-TeX

```
333 ⟨*test3⟩
334 \def\msg#{\immediate\write16}
335 \newlinechar=10 %
336 \input catchfile.sty\relax
337
338 \def\mysetup{%
339   \def\do##1{%
340     \catcode'##1=12\relax
341   }%
342   \dospecials
343 }
344
345 \CatchFileDef\contents{catchfile.sty}\mysetup
346 \msg{\meaning\contents}
347
348 \CatchFileEdef\contents{catchfile.sty}{%
349   \mysetup
350   \def\par{^^J}%
351   \obeylines
352 }
353 \msg{\contents}
354
355 \csname @@end\endcsname
356 \end
357 ⟨/test3⟩
```

4   Installation

4.1   Download

Package.   This package is available on CTAN[1]:

CTAN:macros/latex/contrib/oberdiek/catchfile.dtx The source file.

CTAN:macros/latex/contrib/oberdiek/catchfile.pdf Documentation.

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 TeX Files" (CTAN:tds/tds.pdf). Directories with texmf in their name are usually organized this way.

---

[1]ftp://ftp.ctan.org/tex-archive/

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
test/catchfile-test1.tex → doc/latex/oberdiek/test/catchfile-test1.tex
test/catchfile-test2.tex → doc/latex/oberdiek/test/catchfile-test2.tex
test/catchfile-test3.tex → doc/latex/oberdiek/test/catchfile-test3.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 LaTeX. The `.dtx` chooses its action depending on the format:

plain-TeX: Run `docstrip` and extract the files.

LaTeX: Generate the documentation.

If you insist on using LaTeX for `docstrip` (really, `docstrip` does not need LaTeX), 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 pdfLATEX:

```
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.

[2007/09/09 v1.1]

- Catcode section rewritten.

[2007/11/11 v1.2]

- Use of package pdftexcmds for LUATEX support.

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.