

The hopatch package

Heiko Oberdiek

<heiko.oberdiek at gmail.com>

2012/05/28 v1.2

Abstract

This packages provides a wrapper to various package hooks provided by other packages or classes, but does not define own hooks.

Contents

1 Documentation	1
2 Implementation	3
2.1 Catcodes and package identification	3
2.2 Resources	4
2.3 Package patching	4
3 Test	6
3.1 Catcode checks for loading	6
4 Installation	9
4.1 Download	9
4.2 Bundle installation	9
4.3 Package installation	9
4.4 Refresh file name databases	10
4.5 Some details for the interested	10
5 Catalogue	10
6 References	11
7 History	11
[2011/01/30 v1.0]	11
[2011/06/24 v1.1]	11
[2012/05/28 v1.2]	11
8 Index	11

1 Documentation

Sometimes I want to add code right after a package has been loaded. Examples are bug fixes, adaptations, or added features as needed by package `hyperref`, for instance.

Table 1: After package hooking

Macro	Provider
<code>\AfterPackage</code>	package scrfile [5]
<code>\AtEndOfPackageFile</code>	package filehook [2]
<code>\AtEndPackage</code>	class memoir [4]

Table 2: After begin document hooking

Macro	Provider
<code>\AtBeginDocument</code>	L ^A T _E X’s kernel
<code>\AtEndPreamble</code>	package etoolbox [1]
<code>\AfterEndPreamble</code>	package etoolbox

Unhappily L^AT_EX does not provide this kind of hook. `\AtEndOfPackage` can be used inside the package only, because L^AT_EX clears the hook right before it loads the package.

However, there are already many packages and classes that provide hooks that are executed after the package is loaded, see table 1.

Package `hopatch` can be used without the packages of table 1. But for an early executing right after a package is loaded, one of the following class or packages should be loaded before using `\hopatch@AfterPackage`:

- package filehook
- package scrfile
- class memoir

Therefore I skip writing a new package for hooking into L^AT_EX’s package management and use this package to provide a wrapper to patch a package after it is loaded.

`\hopatch@AfterPackage {⟨package⟩} {⟨patch code⟩}`

If the package is already loaded, the *⟨patch code⟩* is executed immediately. Otherwise the *⟨patch code⟩* is stored in a command and tried at later locations until the package is available.

The patch is tried in the following order:

1. If the package is already loaded, the patch is applied immediately. Further locations are not tried.
2. `\AtEndPackage`, provided by class memoir [4], and `\AfterPackage`, provided by package scrfile [5], are called right after the package file is input before the hook of L^AT_EX’s `\AtEndOfPackage`.
3. `\AtEndOfPackageFile`, provided by package filehook [2], is called after the package is loaded and after the hook of L^AT_EX’s `\AtEndOfPackage`.
4. `\AtEndPreamble`, provided by package etoolbox [1], is called at the beginning of `\begin{document}` before the hook of L^AT_EX’s `\AtBeginDocument`.
5. `\AtBeginDocument`, provided by L^AT_EX.

6. `\AfterEndDocument`, provided by package `etoolbox` [1], is called at the very end of `\begin{document}`. Preamble commands are already forbidden there.

Because of the various locations the patch code is restricted to limitations:

- Preamble commands, see L^AT_EX's `\onlypreamble` throw an error if used after `\begin{document}`. This is already the case for `\AfterEndDocument`. Therefore preamble commands are forbidden in the patching code. There are four exceptions `\ifpackageloaded`, `\ifclassloaded`, `\ifpackageafter` and `\ifclasslater`. They are redefined during `\AfterEndDocument` using the counterparts of package `ltxcmds` [3].
- `\AfterPackage` of package `scrfile` and `\AtEndPackage` of class `memoir` call the hook before L^AT_EX's `\AtEndOfPackage`.

2 Implementation

```
1 \<{*package}
```

2.1 Catcodes and package identification

```
2 \begingroup\catcode61\catcode48\catcode32=10\relax%
3 \catcode13=5 % ^^M
4 \endlinechar=13 %
5 \catcode123=1 % {
6 \catcode125=2 % }
7 \catcode64=11 % @
8 \def\x{\endgroup
9 \expandafter\edef\csname H0patch@AtEnd\endcsname{%
10 \endlinechar=\the\endlinechar\relax
11 \catcode13=\the\catcode13\relax
12 \catcode32=\the\catcode32\relax
13 \catcode35=\the\catcode35\relax
14 \catcode61=\the\catcode61\relax
15 \catcode64=\the\catcode64\relax
16 \catcode123=\the\catcode123\relax
17 \catcode125=\the\catcode125\relax
18 }%
19 }%
20 \x\catcode61\catcode48\catcode32=10\relax%
21 \catcode13=5 % ^^M
22 \endlinechar=13 %
23 \catcode35=6 % #
24 \catcode64=11 % @
25 \catcode123=1 % {
26 \catcode125=2 % }
27 \def\TMP@EnsureCode#1#2{%
28 \edef\H0patch@AtEnd{%
29 \H0patch@AtEnd
30 \catcode#1=\the\catcode#1\relax
31 }%
32 \catcode#1=#2\relax
33 }
34 \TMP@EnsureCode{40}{12}% (
35 \TMP@EnsureCode{41}{12}% )
36 \TMP@EnsureCode{43}{12}% +
37 \TMP@EnsureCode{46}{12}% .
38 \TMP@EnsureCode{47}{12}% /
```

```

39 \TMP@EnsureCode{91}{12}% [
40 \TMP@EnsureCode{93}{12}% ]
41 \edef\H0patch@AtEnd{\H0patch@AtEnd\noexpand\endinput}

```

Package identification.

```

42 \NeedsTeXFormat{LaTeX2e}
43 \ProvidesPackage{hopatch}%
44 [2012/05/28 v1.2 Wrapper for package hooks (H0)]

```

2.2 Resources

```

45 \begingroup\expandafter\expandafter\expandafter\endgroup
46 \expandafter\ifx\csname RequirePackage\endcsname\relax
47 \def\TMP@RequirePackage#1[#2]{%
48 \begingroup\expandafter\expandafter\expandafter\endgroup
49 \expandafter\ifx\csname ver@#1.sty\endcsname\relax
50 \input #1.sty\relax
51 \fi
52 }%
53 \TMP@RequirePackage{ltxcms}[2010/12/12]%
54 \else
55 \RequirePackage{ltxcms}[2010/12/12]%
56 \fi

```

\H0patch@counter

```

57 \def\H0patch@counter{0}%

```

\H0patch@StepCounter

```

58 \ltx@ifundefined{numexpr}{%
59 \def\H0patch@StepCounter{%
60 \begingroup
61 \count@\H0patch@counter\relax
62 \advance\count@\ltx@one\relax
63 \edef\x{\endgroup
64 \noexpand\def\noexpand\H0patch@counter{\the\count@}%
65 }%
66 \x
67 }%
68 }{%
69 \def\H0patch@StepCounter{%
70 \edef\H0patch@counter{%
71 \the\numexpr\H0patch@counter+\ltx@one\relax
72 }%
73 }%
74 }

```

\H0patch@list

```

75 \def\H0patch@list{}

```

\H0patch@Add

```

76 \def\H0patch@Add{%
77 \ltx@LocalAppendToMacro\H0patch@list
78 }

```

2.3 Package patching

\hopatch@AfterPackage

```

79 \def\hopatch@AfterPackage#1{%
80 \ltx@ifpackageloaded{#1}{%

```

```

81     \ltx@firstofone
82   }{%
83     \HOpatch@AfterPackage{#1}%
84   }%
85 }

\HOpatch@AfterPackage

86 \def\HOpatch@AfterPackage#1{%
87   \edef\HOpatch@temp{#1}%
88   \HOpatch@StepCounter
89   \expandafter\HOpatch@@AfterPackage
90   \csname HOpatch@\HOpatch@counter\expandafter\endcsname{%
91     \HOpatch@temp
92   }%
93 }

\HOpatch@@AfterPackage

94 \def\HOpatch@@AfterPackage#1#2#3{%
95   \begingroup
96     \toks@{#3}%
97     \xdef\HOpatch@gtemp{%
98       \noexpand\ltx@ifpackageloaded{#2}{%
99         \noexpand\let\noexpand#1\noexpand\relax
100        \the\toks@
101      }{%}%
102    }%
103   \endgroup
104   \let#1\HOpatch@gtemp
105   \HOpatch@Add#1%
106   \HOpatch@Try{AfterPackage}{#2}#1%
107   \HOpatch@Try{AtEndPackage}{#2}#1%
108   \HOpatch@Try{AtEndOfPackageFile}{#2}#1%
109 }

\HOpatch@Try

110 \def\HOpatch@Try#1#2#3{%
111   \ltx@ifundefined{#1}{}{%
112     \csname #1\endcsname{#2}{#3}%
113   }%
114 }

115 \AtBeginDocument{\HOpatch@list}
116 \ltx@ifundefined{AtEndPreamble}{}{%
117   \ltx@ifundefined{@endpreamblehook}{}{%
118     \AtEndPreamble{\HOpatch@list}%
119   }%
120 }
121 \ltx@ifundefined{AfterEndPreamble}{}{%
122   \ltx@ifundefined{@afterendpreamblehook}{}{%
123     \AfterEndPreamble{%
124       \let\HOpatch@OrgIfPackageLoaded\@ifpackageloaded
125       \let\HOpatch@OrgIfPackageLater\@ifpackagelater
126       \let\HOpatch@OrgIfClassLoaded\@ifclassloaded
127       \let\HOpatch@OrgIfClassLater\@ifclasslater
128       \let\@ifpackageloaded\ltx@ifpackageloaded
129       \let\@ifpackagelater\ltx@ifpackagelater
130       \let\@ifclassloaded\ltx@ifclassloaded
131       \let\@ifclasslater\ltx@ifclasslater

```

```

132     \HOpatch@list
133     \let\@ifpackageloaded\HOpatch@OrgIfPackageLoaded
134     \let\@ifpackagelater\HOpatch@OrgIfPackageLater
135     \let\@ifclassloaded\HOpatch@OrgIfClassLoaded
136     \let\@ifclasslater\HOpatch@OrgIfClassLater
137   }%
138 }%
139 }

140 \HOpatch@AtEnd%
141 \</package>

```

3 Test

```

142 \<test1>
143 \def\LoadCommand{\RequirePackage{hopatch}[2012/05/28]}
144 \</test1>

```

3.1 Catcode checks for loading

```

145 \<test1>
146 \catcode'\{=1 %
147 \catcode'\}=2 %
148 \catcode'\#=6 %
149 \catcode'\@=11 %
150 \expandafter\ifx\csname count@endcsname\relax
151   \countdef\count@=255 %
152 \fi
153 \expandafter\ifx\csname @gobble@endcsname\relax
154   \long\def\@gobble#1{}%
155 \fi
156 \expandafter\ifx\csname @firstofone@endcsname\relax
157   \long\def\@firstofone#1{#1}%
158 \fi
159 \expandafter\ifx\csname loop@endcsname\relax
160   \expandafter\@firstofone
161 \else
162   \expandafter\@gobble
163 \fi
164 {%
165   \def\loop#1\repeat{%
166     \def\body{#1}%
167     \iterate
168   }%
169   \def\iterate{%
170     \body
171     \let\next\iterate
172   \else
173     \let\next\relax
174   \fi
175   \next
176 }%
177 \let\repeat=\fi
178 }%
179 \def\RestoreCatcodes{}
180 \count@=0 %
181 \loop
182   \edef\RestoreCatcodes{%

```

```

183 \RestoreCatcodes
184 \catcode\the\count@=\the\catcode\count@\relax
185 }%
186 \ifnum\count@<255 %
187 \advance\count@ 1 %
188 \repeat
189
190 \def\RangeCatcodeInvalid#1#2{%
191 \count@=#1\relax
192 \loop
193 \catcode\count@=15 %
194 \ifnum\count@<#2\relax
195 \advance\count@ 1 %
196 \repeat
197 }
198 \def\RangeCatcodeCheck#1#2#3{%
199 \count@=#1\relax
200 \loop
201 \ifnum#3=\catcode\count@
202 \else
203 \errmessage{%
204 Character \the\count@\space
205 with wrong catcode \the\catcode\count@\space
206 instead of \number#3%
207 }%
208 \fi
209 \ifnum\count@<#2\relax
210 \advance\count@ 1 %
211 \repeat
212 }
213 \def\space{ }
214 \expandafter\ifx\csname LoadCommand\endcsname\relax
215 \def\LoadCommand{\input hopatch.sty\relax}%
216 \fi
217 \def\Test{%
218 \RangeCatcodeInvalid{0}{47}%
219 \RangeCatcodeInvalid{58}{64}%
220 \RangeCatcodeInvalid{91}{96}%
221 \RangeCatcodeInvalid{123}{255}%
222 \catcode'\@=12 %
223 \catcode'\=0 %
224 \catcode'\%=14 %
225 \LoadCommand
226 \RangeCatcodeCheck{0}{36}{15}%
227 \RangeCatcodeCheck{37}{37}{14}%
228 \RangeCatcodeCheck{38}{47}{15}%
229 \RangeCatcodeCheck{48}{57}{12}%
230 \RangeCatcodeCheck{58}{63}{15}%
231 \RangeCatcodeCheck{64}{64}{12}%
232 \RangeCatcodeCheck{65}{90}{11}%
233 \RangeCatcodeCheck{91}{91}{15}%
234 \RangeCatcodeCheck{92}{92}{0}%
235 \RangeCatcodeCheck{93}{96}{15}%
236 \RangeCatcodeCheck{97}{122}{11}%
237 \RangeCatcodeCheck{123}{255}{15}%
238 \RestoreCatcodes
239 }
240 \Test

```

```

241 \csname @@end\endcsname
242 \end
243 </test1>
244 <*test2>
245 \NeedsTeXFormat{LaTeX2e}
246 \providecommand\variant{0}
247 \RequirePackage{filecontents}
248 \begin{filecontents}{foo.sty}
249 \ProvidesPackage{foo}
250 \def\msg#\immediate\write16{
251 \def\foo#1{%
252   \msg{\fooformat{#1}}%
253 }
254 \def\fooformat#1{[#1]}% hash-ok
255 \foo{* Executing foo at package loading}
256 \end{filecontents}
257
258 \ifnum\variant=1 %
259   \documentclass{memoir}%
260 \else
261   \documentclass{article}%
262 \fi
263
264 \ifcase\variant\relax
265 \or % 1
266 \or % 2
267   \usepackage{etoolbox}%
268 \or % 3
269   \usepackage{scrfile}%
270 \or % 4
271   \usepackage{filehook}%
272 \fi
273
274 \AtBeginDocument{\foo{* AtBeginDocument before hopatch}}
275 \usepackage{hopatch}
276 \AtBeginDocument{\foo{* AtBeginDocument after hopatch}}
277
278 \makeatletter
279 \hopatch@AfterPackage{foo}{%
280   \def\fooformat#1{<<#1>>}%
281 }
282 \makeatother
283
284 \AtBeginDocument{\foo{* AtBeginDocument before foo}}
285 \usepackage{foo}
286 \AtBeginDocument{\foo{* AtBeginDocument after foo}}
287
288 \foo{* Executing in preamble}
289
290 \begin{document}
291 \foo{* Executing in document}
292 \end{document}
293 </test2>

```


4 Installation

4.1 Download

Package. This package is available on CTAN¹:

[CTAN:macros/latex/contrib/oberdiek/hopatch.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/hopatch.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.

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 hopatch.dtx
```

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

<code>hopatch.sty</code>	\rightarrow <code>tex/latex/oberdiek/hopatch.sty</code>
<code>hopatch.pdf</code>	\rightarrow <code>doc/latex/oberdiek/hopatch.pdf</code>
<code>test/hopatch-test1.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/hopatch-test1.tex</code>
<code>test/hopatch-test2.tex</code>	\rightarrow <code>doc/latex/oberdiek/test/hopatch-test2.tex</code>
<code>hopatch.dtx</code>	\rightarrow <code>source/latex/oberdiek/hopatch.dtx</code>

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`.

¹<http://ftp.ctan.org/tex-archive/>

4.4 Refresh file name databases

If your $\text{T}_{\text{E}}\text{X}$ distribution ($\text{t}_{\text{E}}\text{X}$, $\text{m}_{\text{ik}}\text{T}_{\text{E}}\text{X}$, ...) relies on file name databases, you must refresh these. For example, $\text{t}_{\text{E}}\text{X}$ users run `texhash` or `mktextlsr`.

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 hopatch.pdf unpack_files output .
```

Unpacking with $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. The `.dtx` chooses its action depending on the format:

plain $\text{T}_{\text{E}}\text{X}$: Run `docstrip` and extract the files.

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$: Generate the documentation.

If you insist on using $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ for `docstrip` (really, `docstrip` does not need $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{hopatch.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 `pdf $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$` :

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

5 Catalogue

The following XML file can be used as source for the [\$\text{T}_{\text{E}}\text{X}\$ Catalogue](#). The elements `caption` and `description` are imported from the original XML file from the Catalogue. The name of the XML file in the Catalogue is `hopatch.xml`.

```
294 (*catalogue)
295 <?xml version='1.0' encoding='us-ascii'?>
296 <!DOCTYPE entry SYSTEM 'catalogue.dtd'>
297 <entry datestamp='$Date$' modifier='$Author$' id='hopatch'>
298   <name>hopatch</name>
299   <caption>Load patches for packages.</caption>
300   <authorref id='auth:oberdiek'>/>
301   <copyright owner='Heiko Oberdiek' year='2011,2012'>/>
302   <license type='lppl1.3'>/>
303   <version number='1.2'>/>
304   <description>
305     The hopatch package provides a command with which the user may
```

```

306   register a piece of patch code for a particular package. Hopatch
307   will apply the patch immediately, if the relevant package has
308   already been loaded; otherwise it will store the patch until the
309   package appears.
310   <p/>
311   The package is part of the <xref refid='oberdiek'>oberdiek</xref> bundle.
312 </description>
313 <documentation details='Package documentation'
314   href='ctan:/macros/latex/contrib/oberdiek/hopatch.pdf' />
315 <ctan file='true' path='/macros/latex/contrib/oberdiek/hopatch.dtx' />
316 <miktex location='oberdiek' />
317 <texlive location='oberdiek' />
318 <install path='/macros/latex/contrib/oberdiek/oberdiek.tds.zip' />
319 </entry>
320 </catalogue>

```

6 References

- [1] Philipp Lehman: *The etoolbox Package* 2011-01-03.
[CTAN:macros/latex/contrib/etoolbox/etoolbox.pdf](#)
- [2] Martin Scharrer: *The filehook Package*; 2011-01-09.
[CTAN:macros/latex/contrib/filehook/filehook.pdf](#)
- [3] Heiko Oberdiek: *The ltxcmds Package*; 2010-12-12.
[CTAN:macros/latex/contrib/oberdiek/ltxcmds.pdf](#)
- [4] Peter Wilson, Lars Madsen: *The Memoir Class for Configurable Typesetting, User Guide*; 2010. [CTAN:macros/latex/contrib/memoir/memman.pdf](#)
- [5] Markus Kohm, Jens-Uwe Morawski: *The Guide KOMA-Script*; 2011-01-20.
[CTAN:macros/latex/contrib/koma-script/scrguien.pdf](#)

7 History

[2011/01/30 v1.0]

- First public version.

[2011/06/24 v1.1]

- Fix the use of `\AtEndPreamble` and `\AfterEndPreamble`. They are redefined by package `etoolbox` after their hooks are used and generate an error message then.

[2012/05/28 v1.2]

- Fix for use without ε -TeX (thanks Gordon Lee).

8 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; plain numbers refer to the code lines where the entry is used.

Symbols

`\#` [148](#)

<code>\the</code>	10, 11, 12, 13, 14, 15, 16, 17, 30, 64, 71, 100, 184, 204, 205	V	
<code>\TMP@EnsureCode</code>		<code>\variant</code>	246, 258, 264
.	27, 34, 35, 36, 37, 38, 39, 40	W	
<code>\TMP@RequirePackage</code>	47, 53	<code>\write</code>	250
<code>\toks@</code>	96, 100		
	U	X	
<code>\usepackage</code> . . .	267, 269, 271, 275, 285	<code>\x</code>	8, 20, 63, 66