iexec: LaTeX Package for Inputable Shell Executions[*]

Yegor Bugayenko
yegor256@gmail.com

2022-11-01, 0.11.4

NB! This package doesn't work on Windows!

1   Introduction

This package helps you execute shell commands right from the document and then put their output to the document:

Today is **1-Nov-2022**

```
1 \documentclass{article}
2 \usepackage{iexec}
3 \usepackage[paperwidth=3in]{geometry}
4 \pagestyle{empty}
5 \begin{document}
6 Today is \textbf{%
7   \iexec{date +\%e-\%b-\%Y}}
8 \end{document}
```

\iexec   The only command provided by this package is \iexec [⟨*options*⟩] {⟨*cmd*⟩}. Its only mandatory argument ⟨*cmd*⟩ is the command to be executed through the terminal shell (bash, or whatever is set as the default one in your console).

You have to run pdflatex (or just latex) with the --shell-escape flag in order to let shellesc execute your shell command.

2   Options

quiet   If you don't want the output to be visible, use \phantom\{\iexec{...}}. Otherwise, you can use quiet option:

```
I just want to delete some file:
\iexec[quiet]{rm -f foo.txt}
```

In this case, whatever the shell command produces will not be included into the document.

stdout   The output of your code is saved into the file provided as an optional argument of \iexec (the default value is iexec.tmp):

```
Today is \iexec[stdout=date.txt]{date +\%e-\%b-\%Y | tr -d '\\n'}.
```

---

[*]The sources are in GitHub at yegor256/iexec

The tailing part of the command here removes all ends-of-line.

stderr The error output of the code is saved into the file provided as an optional argument of \iexec (by default the error output is streamed into stdout):

```
Today is \iexec[stderr=my.txt]{broken-command}.
```

exit The exit code of the command it saved into a file. You can change the name of it using exit option:

```
Today is \iexec[exit=code.txt]{./broken-command.sh}.
```

trace The file specified will be deleted right after its usage. If you don't want this to happen, use trace package option: all files will remain in the directory where they were created. It's possible to turn tracing on globbaly, for the entire document, using trace option of the package:

```
\documentclass{article}
\usepackage[trace]{iexec}
\begin{document}
This file won't be deleted: \iexec[stdout=me.txt]{whoami}.
\end{document}
```

append The stdout produced will be appended to the file specified:

```
\documentclass{article}
\usepackage[trace]{iexec}
\begin{document}
\iexec[append,stdout=foo.txt,quiet]{echo 'Hello, '}
\iexec[append,stdout=foo.txt,quiet]{echo 'Jeffrey!'}
\input{foo.txt}
\end{document}
```

log The stdout produced will be printed in TeX log:

```
\iexec[log]{echo 'Hello, \\LaTeX!'}
```

null The stdout of the command will be sent to /dev/null:

```
\iexec[null]{rm some-file.txt}
```

ignore By default, we report an error if exit code is not equal to zero. You can suppress this with ignore option:

```
\iexec[ignore]{broken-command}
```

3   Implementation

First, we include shellesc package, which we use to execute shell commands:

```
1 \RequirePackage{shellesc}
```

Then, we parse package options:

```
2 \RequirePackage{xkeyval}
3 \makeatletter
4 \newif\ifiexec@trace
5 \DeclareOptionX{trace}{\iexec@tracetrue}
6 \ProcessOptionsX\relax
7 \makeatother
```

Then, we prepare to parse the options of \iexec command:

```
 8 \RequirePackage{pgfkeys}
 9 \makeatletter\pgfkeys{
10   /iexec/.is family,
11   /iexec,
12   exit/.store in = \iexec@exit,
13   exit/.default = iexec.ret,
14   stdout/.store in = \iexec@stdout,
15   stdout/.default = iexec.tmp,
16   stderr/.store in = \iexec@stderr,
17   trace/.store in = \iexec@traceit,
18   append/.store in = \iexec@append,
19   log/.store in = \iexec@log,
20   null/.store in = \iexec@null,
21   quiet/.store in = \iexec@quiet,
22   ignore/.store in = \iexec@ignore,
23   stdout,exit
24 }\makeatother
```

\iexec@typeout Then, we define an internal command \iexec@typeout for printing the content of a file, as suggested [here](#):

```
25 \RequirePackage{expl3}
26 \makeatletter\ExplSyntaxOn
27 \NewDocumentCommand{\iexec@typeout}{m}{
28   \iexec_typeout_file:n { #1 }}
29 \ior_new:N \g_iexec_typeout_ior
30 \cs_new_protected:Nn \iexec_typeout_file:n
31 {
32   \ior_open:Nn \g_iexec_typeout_ior { #1 }
33   \ior_str_map_inline:Nn \g_iexec_typeout_ior
34     {\iow_term:n { ##1 }}
35   \ior_close:N \g_iexec_typeout_ior
36 }
37 \ExplSyntaxOff\makeatother
```

\iexec Then, we define \iexec command. It is implemented with the help of \ShellEscape from shellesc package:

```
38 \makeatletter
39 \newread\iexec@exitfile
40 \newcommand\iexec[2][]{%
41   \begingroup%
42     \pgfqkeys{/iexec}{#1}%
```

First, we verify that latex is running with --shell-escape option, since without it nothing will work; so, it's better to throw an error earlier than later:

```
43     \ifnum\ShellEscapeStatus=1\else%
44       \PackageError{iexec}{You must run TeX processor with
45       --shell-escape option}{}%
46     \fi%
47     \begingroup%
```

Then, we start the log from a clean line:

```
48       \ifdefined\iexec@log%
49         \message{^^J}%
50       \fi%
```

Then, we define a few special chars in order to escape them in the shell (the full list of them is in macros2e):

```
51        \let\%\@percentchar%
52        \let\\\@backslashchar%
53        \let\{\@charlb%
54        \let\}\@charrb%
```

Then, we execute it and save exit code into a file (where we also add % in order to trim the content to exactly one number, as suggested here):

```
55        \def\iexec@cmd{(#2)
56          \ifdefined\iexec@append>\fi>
57          \ifdefined\iexec@null/dev/null\else\iexec@stdout\fi
58          \space\ifdefined\iexec@stderr2>\iexec@stderr\else2>&1\fi;
59          /bin/echo -n \string$?\% >\iexec@exit}%
60        \ShellEscape{\iexec@cmd}%
```

Then, a message is printed to TEX log:

```
61        \ifdefined\iexec@log%
62          \message{iexec: [\iexec@cmd]^^J}%
63        \fi%
64      \endgroup%
```

Then, we read back the exit code, from the file:

```
65        \immediate\openin\iexec@exitfile=\iexec@exit%
66        \read\iexec@exitfile to \iexec@code%
67        \immediate\closein\iexec@exitfile%
```

Then, if required, we print the content of the stdout file to TEX log:

```
68        \ifdefined\iexec@null\else%
69        \ifdefined\iexec@log%
70          \message{iexec: This is the content of '\iexec@stdout':^^J}%
71          \iexec@typeout{\iexec@stdout}%
72          \message{<EOF>^^J}%
73        \else%
74          \ifnum\iexec@code=0\else%
75            \ifdefined\iexec@ignore\else%
76              \message{iexec: See the content of '\iexec@stdout'
77                after failure:^^J}%
78              \iexec@typeout{\iexec@stdout}%
79              \message{<EOF>^^J}%
80            \fi%
81          \fi%
82        \fi\fi%
```

Then, we check whether it's zero or not (if not zero, we either print a message or fail the build, depending on the presence of ignore option):

```
83        \ifnum\iexec@code=0\else%
84          \ifdefined\iexec@ignore%
85            \ifdefined\iexec@log%
86              \message{iexec: Execution failure ignored,
87                the exit code was \iexec@code^^J}%
88            \fi%
89          \else%
90            \PackageError{iexec}{Execution failure,
91              the exit code was \iexec@code}{}%
92          \fi%
```

```latex
93    \fi%
```

Then, we include the produced output into the current document:

```latex
 94    \ifdefined\iexec@null\else%
 95    \ifdefined\iexec@quiet%
 96      \ifdefined\iexec@log%
 97        \message{iexec: Due to 'quiet' option we didn't read
 98        the content of '\iexec@stdout'
 99        \ifdefined\pdffilesize (\pdffilesize{\iexec@stdout}
100        bytes)\fi^^J}%
101      \fi%
102    \else%
103      \ifdefined\iexec@log%
104        \message{iexec: We are going to include the content of
105        '\iexec@stdout'\ifdefined\pdffilesize (\pdffilesize
106        {\iexec@stdout} bytes)\fi...^^J}%
107      \fi%
108      \input{\iexec@stdout}%
109      \message{iexec: The content of '\iexec@stdout'
110      was included into the document^^J}%
111    \fi\fi%
```

Finally, we delete the file or leave it untouched:

```latex
112    \ifdefined\iexec@null\else%
113    \ifiexec@trace%
114      \ifdefined\iexec@log%
115        \message{iexec: Due to package option 'trace',
116        the files '\iexec@stdout' and '\iexec@exit` were
117        not deleted^^J}%
118      \fi%
119    \else%
120      \ifdefined\iexec@traceit%
121        \ifdefined\iexec@log%
122          \message{iexec: Due to 'trace' package option,
123          the files '\iexec@stdout' and '\iexec@exit'
124          were not deleted^^J}%
125        \fi%
126      \else%
127        \ShellEscape{rm \iexec@stdout}%
128        \ifdefined\iexec@log%
129          \message{iexec: The file '\iexec@stdout' was deleted^^J}%
130        \fi%
131        \ShellEscape{rm \iexec@exit}%
132        \ifdefined\iexec@log%
133          \message{iexec: The file '\iexec@exit' was deleted^^J}%
134        \fi%
135      \fi%
136    \fi\fi%
137  \endgroup%
138 }\makeatother
```

Change History

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.