Package ‘littler’

October 27, 2019

**Type**  Package

**Title**  R at the Command-Line via ’r’

**Version**  0.3.9

**Date**  2019-10-27

**Author**  Dirk Eddelbuettel and Jeff Horner

**Maintainer**  Dirk Eddelbuettel <edd@debian.org>

**Description**  A scripting and command-line front-end

is provided by ’r’ (aka ’littler’) as a lightweight binary wrapper around
the GNU R language and environment for statistical computing and graphics.
While R can be used in batch mode, the r binary adds full support for
both 'shebang'-style scripting (i.e. using a hash-mark-exclamation-path
expression as the first line in scripts) as well as command-line use in
standard Unix pipelines. In other words, r provides the R language without
the environment.

**URL**  [http://dirk.eddelbuettel.com/code/littler.html](http://dirk.eddelbuettel.com/code/littler.html)

**BugReports**  [https://github.com/eddelbuettel/littler/issues](https://github.com/eddelbuettel/littler/issues)

**License**  GPL (>= 2)

**OS_type**  unix

**SystemRequirements**  libR

**Suggests**  knitr, docopt, rcmdcheck

**VignetteBuilder**  knitr

**RoxygenNote**  5.0.1

**NeedsCompilation**  yes

**Repository**  CRAN

**Date/Publication**  2019-10-27 13:57:31

**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>littler</td>
<td>2</td>
</tr>
<tr>
<td>r</td>
<td>3</td>
</tr>
</tbody>
</table>
Description

The `r` binary provides a convenient and powerful front-end. By embedding R, it permits four distinct ways to leverage the power of R at the shell prompt: scripting, filename execution, piping and direct expression evaluation.

Details

The `r` front-end was written with four distinct usage modes in mind.

First, it allow to write so-called ‘shebang’ scripts starting with `#!/usr/bin/env r`. These ‘shebang’ scripts are perfectly suited for automation and execution via e.g. via `cron`.

Second, we can use `r somefile.R` to quickly execute the name R source file. This is useful as `r` is both easy to type—and quicker to start that either `R` itself, or its scripting tool `Rscript`, while still loading the methods package.

Third, `r` can be used in ‘pipes’ which are very common in Unix. A simple and trivial example is `echo 'cat(2+2)' | r` illustrating that the standard output of one program can be used as the standard input of another program.

Fourth, `r` can be used as a calculator by supplying expressions after the `-e` or `--eval` options.

Value

Common with other shell tools and programs, `r` returns its exit code where a value of zero indicates success.

Note

On OS X one may have to link the binary to, say, `lr` instead. As OS X insists that files named `R` and `r` are the same, we cannot use the latter.

Author(s)

Jeff Horner and Dirk Eddelbuettel wrote `littler` from 2006 to today, with contributions from several others.

Dirk Eddelbuettel `<edd@debian.org>` is the maintainer.

Examples

```r
## Not run:
#!/usr/bin/env r ## for use in scripts
other input | r ## for use in pipes
```
r

r somefile.R        ## for running files
r -e 'expr'        ## for evaluating expressions
r --help         ## to show a quick synopsis

## End(Not run)

r Return Path to r Binary

Description
Return the path of the install r binary.

Usage
r(usecat = FALSE)

Arguments
usecat                Optional toggle to request output to stdout (useful in Makefiles)

Details
The test for Windows is of course superfluous as we have no binary for Windows. Maybe one day...

Value
The path is returned as character variable. If the usecat option is set the character variable is displayed via cat instead.

Author(s)
Dirk Eddelbuettel
Index

*Topic package
  littler, 2

cat, 3
littler, 2
r, 3