Package ‘littler’

October 6, 2021

**Type**  Package

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

**Version**  0.3.14

**Date**  2021-10-05

**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**  https://github.com/eddelbuettel/littler,
         https://dirk.eddelbuettel.com/code/littler.html,
         https://eddelbuettel.github.io/littler/

**BugReports**  https://github.com/eddelbuettel/littler/issues

**License**  GPL (>= 2)

**OS_type**  unix

**SystemRequirements**  libR

**Suggests**  knitr, rmarkdown, minidown, docopt, rcmdcheck, foghorn

**VignetteBuilder**  knitr

**RoxygenNote**  5.0.1

**NeedsCompilation**  yes

**Repository**  CRAN

**Date/Publication**  2021-10-05 23:30:02 UTC
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. `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 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
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

* package
  littler, 2

cat, 3
littler, 2
r, 3