Package 'startup'

October 16, 2022

Version 0.19.0

Title Friendly R Startup Configuration

Depends R (>= 2.14.0)

Suggests commonmark, tools, tcltk

VignetteBuilder startup

Description Adds support for R startup configuration via '.Renviron.d' and '.Rprofile.d' directories in addition to '.Renviron' and '.Rprofile' files. This makes it possible to keep private / secret environment variables separate from other environment variables. It also makes it easier to share specific startup settings by simply copying a file to a directory.

License LGPL (>= 2.1)

LazyLoad TRUE

URL https://henrikbengtsson.github.io/startup/, https://github.com/HenrikBengtsson/startup

BugReports https://github.com/HenrikBengtsson/startup/issues

RoxygenNote 7.2.1

NeedsCompilation no

Author Henrik Bengtsson [aut, cre, cph]

Maintainer Henrik Bengtsson <henrikb@braju.com>

Repository CRAN

Date/Publication 2022-10-16 02:50:02 UTC

R topics documented:

---

check *Check for and fix common mistakes in .Rprofile*

Description

Check for and fix common mistakes in '`.Rprofile`' files.

Usage

```
check(all = FALSE, fix = TRUE, backup = TRUE, debug = FALSE)
```

Arguments

`all`	Should all or only the first entry on the R startup search path be checked?
`fix`	If `TRUE`, detected issues will be tried to be automatically fixed, otherwise not.
`backup`	If `TRUE`, a timestamped backup copy of the original file is created before modifying it, otherwise not.
`debug`	If `TRUE`, debug messages are outputted, otherwise not.

Value

Returns invisibly a character vector of files that were "fixed" (modified), if any. If no files needed to be fixed, or `fix = TRUE`, then an empty vector is returned.

References

1. R-devel thread 'Last line in .Rprofile must have newline (PR#4056)', 2003-09-03, https://stat.ethz.ch/pipermail/r-devel/2003-September/027457.html

current_script *Gets the pathname of the currently running startup script*

Description

Gets the pathname of the currently running startup script

Usage

```
current_script()
```

Value

A character string

install *Install and uninstall support for .Renviron.d and .Rprofile.d startup directories*

Description

Install and uninstall support for '.Renviron.d' and '.Rprofile.d' startup directories by appending / removing one line of code to the '~/.Rprofile' file.

Usage

```
install(
  path = "~",
  backup = TRUE,
  overwrite = FALSE,
  make_dirs = TRUE,
  quiet = FALSE
)

uninstall(path = "~", backup = TRUE, quiet = FALSE)
```

Arguments

path	The path where to create / update the '.Rprofile' file.
backup	If TRUE, a timestamped backup copy of the original file is created before modifying / overwriting it, otherwise not. If the backup fails, then an error is produced and the R startup file is unmodified.
overwrite	If the R startup file already exist, then FALSE (default) appends the startup code to the end of the file. is overwritten. If TRUE, any pre-existing R startup file is overwritten.

make_dirs	If TRUE (default), directories '.Renviron.d/' and '.Rprofile.d/' are created in folder path, if missing.
quiet	If FALSE (default), detailed messages are generated, otherwise not.

Value

The pathname of the R startup file modified.

Functions

- install(): injects a tryCatch(startup::startup(), ...) call to the '.Rprofile' file, which is created if missing.
- uninstall(): Remove calls to startup::startup() and similar.

---

is_ess *Checks whether running R via Emacs Spekas Statistics (ESS) or not*

Description

Checks whether running R via Emacs Spekas Statistics (ESS) or not

Usage

```
is_ess()
```

Value

A logical

---

is_microsoftr *Checks if running R in Microsoft R Open*

Description

Checks if running R in Microsoft R Open

Usage

```
is_microsoftr()
```

Value

A logical

is_pqr *Checks if running pqR (A Pretty Quick Version of R)*

Description

Checks if running pqR (A Pretty Quick Version of R)

Usage

```
is_pqr()
```

Value

A logical

References

1. pqR - a pretty quick version of R, http://www.pqr-project.org/
2. GitHub repository for 'pqR' https://github.com/radfordneal/pqR

is_radian *Checks whether running R via radian (formerly known as rtichoke and rice) or not*

Description

Checks whether running R via radian (formerly known as rtichoke and rice) or not

Usage

```
is_radian()
```

Value

A logical

References

1. radian - A 21 century R console (previously known as rtichoke and rice), https://github.com/randy3k/radian

is_rstudio_console *Checks if running R in RStudio Console or RStudio Terminal*

Description

Checks if running R in RStudio Console or RStudio Terminal

Usage

```
is_rstudio_console()
```

Value

A logical

References

1. Kevin Ushey (RStudio), Check if R is running in RStudio, Stackoverflow, 2016-09-23, https://stackoverflow.com/questions/12389158/check-if-r-is-running-in-rstudio#comment66636507_35849779
2. Jonathan McPherson (RStudio), Programmatically detect RStudio Terminal vs RStudio Console?, RStudio Community - RStudio IDE, 2018-01-10, https://community.rstudio.com/t/programmatically-detect-rstudio-terminal-vs-rstudio-console/4107

is_wine *Checks whether running R on Windows using Wine or not*

Description

Checks whether running R on Windows using Wine or not

Usage

```
is_wine()
```

Value

A logical

References

1. Wine Developer FAQ, How can I detect Wine?, https://wiki.winehq.org/Developer_FAQ#How_can_I_detect_Wine.3F
2. Jeff Zaroyko, Detecting Wine, Wine Devel mailing list, 2008-09-29, https://www.winehq.org/pipermail/wine-devel/2008-September/069387.html

on_session_enter *Register functions to be evaluated at the beginning or end of the R session*

Description

Register functions to be evaluated at the beginning or end of the R session

Usage

```
on_session_enter(fcn = NULL, append = TRUE, replace = FALSE)

on_session_exit(fcn = NULL, append = TRUE, replace = FALSE)
```

Arguments

fcn	A function or an R expression. The function must accept zero or more arguments (currently not used). If an expression, it will automatically we wrapped up in an anonymous function.
append	If TRUE (default), the function will be evaluated after previously registered ones, otherwise prepended.
replace	if TRUE, the function replaces any previously registered ones, otherwise it will be added (default).

Details

These functions register one or more functions to be called when the current R session begins or ends. The functions are evaluated in a local environment and without exception handlers, which means that if one produces an error, then none of the succeeding functions will be called.

To list currently registered functions, use `fcns <- on_session_enter()` or `fcns <- on_session_exit()`. To remove all registered functions, use `on_session_enter(replace = TRUE)` or `on_session_exit(replace = TRUE)`.

The `on_session_enter()` function works by recording all `fcn`:s in an internal list which will be evaluated via a custom `.First()` function created in the global environment. Any other `.First()` function on the search path, including a pre-existing `.First()` function in the global environment, is called at the end after registered functions have been called.

The `on_session_exit()` function works by recording all `fcn`:s in an internal list which will be evaluated via a custom function that is called when the global environment is garbage collected, which happens at the very end of the R shutdown process. Contrary to a `.Last()` function, which is not be called if `quit(runLast = FALSE)` is used, functions registered via `on_session_exit()` are always processed. Registered `on_session_exit()` functions are called *after* `quit()` saves any workspace image to file (`./.RData`), and *after* any `.Last()` has been called.

Value

(invisible) the list of registered functions.

Examples

```
## Not run:
## Summarize interactive session upon termination
if (interactive()) {
  startup::on_session_exit(local({
    t0 <- Sys.time()
    function(...) {
      dt <- difftime(Sys.time(), t0, units = "auto")
      msg <- c(
        "Session summary:",
        sprintf(" * R version: %s", getRversion()),
        sprintf(" * Process ID: %d", Sys.getpid()),
        sprintf(" * Wall time: %.2f %s", dt, attr(dt, "units"))
      )
      msg <- paste(msg, collapse = "\n")
      message(msg)
    }
  }))
}

## End(Not run)
```

`renviron_d` *Load .Renviron.d and .Rprofile.d directories during the R startup process*

Description

Initiates R using all files under '`.Renviron.d/`' and / or '`.Rprofile.d/`' directories (or in subdirectories thereof).

Usage

```
renviron_d(
  sibling = FALSE,
  all = FALSE,
  unload = FALSE,
  skip = NA,
  dryrun = NA,
  debug = NA,
  paths = NULL
)

rprofile_d(
  sibling = FALSE,
  all = FALSE,
  check = NA,
```

```
  unload = FALSE,
  skip = NA,
  on_error = c("error", "warning", "immediate.warning", "message", "ignore"),
  dryrun = NA,
  debug = NA,
  paths = NULL
)

startup(
  sibling = FALSE,
  all = FALSE,
  on_error = c("error", "warning", "immediate.warning", "message", "ignore"),
  keep = c("options"),
  check = NA,
  unload = TRUE,
  skip = NA,
  dryrun = NA,
  debug = dryrun
)
```

Arguments

sibling	If TRUE, then only '.Renviron.d/' and '.Rprofile.d/' directories with a sibling '.Renviron' and '.Rprofile' in the same location will be considered.
all	If TRUE, then *all* '.Renviron.d/' and '.Rprofile.d/' directories found on the R startup search path are processed, otherwise only the *first ones* found.
unload	If TRUE, then the package is unloaded afterward, otherwise not.
skip	If TRUE, startup directories will be skipped. If NA, they will be skipped if command-line options --vanilla, --no-init-file, and / or --no-environ were specified.
dryrun	If TRUE, everything is done except the processing of the startup files.
debug	If TRUE, debug messages are outputted, otherwise not.
paths	(internal) character vector of directories.
check	If TRUE, then the content of startup files are validated.
on_error	Action taken when an error is detected when sourcing an Rprofile file. It is not possible to detect error in Renviron files; they are always ignored with a message that cannot be captured.
keep	Specify what information should remain after this function complete. The default is to keep startup.session.* options as recorded by startup_session_options().

Details

The above is done in addition the '.Renviron' and '.Rprofile' files that are supported by the built-in startup process of R.

Functions

- `renviron_d()`: Initiate using '`.Renviron.d/`' files
- `rprofile_d()`: Initiate using '`.Rprofile.d/`' files
- `startup()`: `renviron_d()` followed by `rprofile_d()` and then the package is unloaded

User-specific installation

In order for '`.Rprofile.d`' and '`.Renviron.d`' directories to be included during the R startup process, a user needs to add `startup::startup()` to '`~/.Rprofile`'. Adding this can also be done by calling `install()` once.

Site-wide installation

An alternative to having each user add `startup::startup()` in their own '`~/.Rprofile`' file, is to add it to the site-wide '`Rprofile.site`' file (see ?Startup). The advantage of such a site-wide installation, is that the users do not have to have a '`.Rprofile`' file for '`.Rprofile.d`' and '`.Renviron.d`' directories to work. For this to work for all users automatically, the **startup** package should also be installed in the site-wide library.

Examples

```
## Not run:
# The most common way to use the package is to add
# the following call to the ~/.Rprofile file.
startup::startup()

# To process ~/.Renviron.d/ files, and then any ./.Renviron.d/ files,
# followed by  ~/.Rprofile.d/ files, and then any ./.Rprofile.d/ files,
# add the following call to the ~/.Rprofile file.
startup::startup(all = TRUE)

# For finer control of on exactly what files are used
# functions renviron_d() and rprofile_d() are also available:

# Initiate first .Renviron.d/ found on search path
startup::renviron_d()

# Initiate all .Rprofile.d/ directories found on the startup search path
startup::rprofile_d(all = TRUE)

## End(Not run)
```

---

reset_when_cache *Reset all or parts of the "when" cache*

Description

Reset all or parts of the "when" cache

Usage

```
reset_when_cache(
  when = c("once", "hourly", "daily", "weekly", "fortnightly", "monthly")
)
```

Arguments

when	(character vector) Specifies for which "when" frequencies the cache should be reset. The default is to reset all of them.

Value

(invisible) The pathnames of the "when" cache files removed.

---

restart *Restarts R*

Description

Restarts R by quitting the current R session and launching a new one.

Usage

```
restart(
  status = 0L,
  workdir = NULL,
  rcmd = NULL,
  args = NULL,
  envvars = NULL,
  as = c("current", "specified", "R CMD build", "R CMD check", "R CMD INSTALL"),
  quiet = FALSE,
  debug = NA
)
```

Arguments

status	An integer specifying the exit code of the current R session.
workdir	The working directory where the new R session should be launched from. If NULL, then the working directory that was in place when the **startup** package was first loaded. If using `startup::startup()` in an '.Rprofile' startup file, then this is likely to record the directory from which R itself was launched from.
rcmd	A character string specifying the command for launching R. The default is the same as used to launch the current R session, i.e. `commandArgs()[1]`.
args	A character vector specifying zero or more command-line arguments to be appended to the system call of rcmd.
envvars	A named character vector of environment variables to be set when calling R.

as	A character string specifying a predefined setups of rcmd, args, and envvars. For details, see below.
quiet	Should the restart be quiet or not? If NA and as == "current", then quiet is TRUE if the current R session was started quietly, otherwise FALSE.
debug	If TRUE, debug messages are outputted, otherwise not.

Predefined setups

Argument as may take the following values:

"current": (Default) A setup that emulates the setup of the current R session as far as possible by relaunching R with the same command-line call (= base::commandArgs()).

"specified": According to rcmd, args, and envvars.

"R CMD build": A setup that emulates R CMD build as far as possible.

"R CMD check": A setup that emulates R CMD check as far as possible, which happens to be identical to the "R CMD build" setup.

"R CMD INSTALL": A setup that emulates R CMD INSTALL as far as possible.

If specified, command-line arguments in args and environment variables in envvars are *appended* accordingly.

Known limitations

It is *not* possible to restart an R session running in *RGui* on Microsoft Windows using this function.

It is *not* possible to restart an R session in the RStudio *Console* using this function. However, it does work when running R in the RStudio *Terminal*.

RStudio provides rstudioapi::restartSession() which will indeed restart the RStudio Console. However, it does not let you control how R is restarted, e.g. with what command-line options and what environment variables. Importantly, the new R session will have the same set of packages loaded as before, the same variables in the global environment, and so on.

Examples

```
## Not run:
  ## Relaunch R with debugging of startup::startup() enabled
  startup::restart(envvars = c(R_STARTUP_DEBUG = TRUE))

  ## Relaunch R without loading user Rprofile files
  startup::restart(args = "--no-init-file")

  ## Mimic 'R CMD build' and 'R CMD check'
  startup::restart(as = "R CMD build")
  startup::restart(as = "R CMD check")
  ## ... which are both short for
  startup::restart(args = c("--no-restore"),
                   envvars = c(R_DEFAULT_PACKAGES="", LC_COLLATE="C"))

## End(Not run)
```

`startup.options` *Options and environment variables used by the 'startup' package*

Description

Below are environment variables and R options that are used by the **startup** package. The `R_STARTUP_***` environment variables must be set before calling the `startup::startup()` function, that is, either (i) prior to launching R or (ii) in the '`.Renviron`' file.

Controls whether startup is used or not

`R_STARTUP_DISABLE` / **'`startup.disable`':** (logical) If `TRUE`, `startup::startup()` is fully disable such that *no* '`.Renviron.d/`' or '`.Rprofile.d/`' files are processed. *Note*: Files '`.Renviron`' and '`.Rprofile`' are still processed because these are out of control of the **startup** package. (Default: `FALSE`)

`R_STARTUP_DRYRUN` / **'`startup.dryrun`':** (logical) Controls the default value of argument `dryrun` of `startup()`. (Default: `FALSE`)

Additional customization of the startup process

`R_STARTUP_FILE` / **'`startup.file`':** (R script as a character string) Optional R script that is parsed and evaluated after '`.Renviron.d/`' and '`.Rprofile.d/`' files, and `R_STARTUP_INIT` code, have been processed, e.g. `R_STARTUP_FILE="setup.R" R --quiet`. (Default: not specified)

`R_STARTUP_INIT` / **'`startup.init`':** (R code as a character string) Optional R code that is parsed and evaluated after '`.Renviron.d/`' and '`.Rprofile.d/`' files, but before `R_STARTUP_FILE` code, have been processed e.g. `R_STARTUP_INIT="message('Hello world')" R --quiet`. The specified string must be parsable by `base::parse()`. (Default: not specified)

`R_STARTUP_RDATA` / **'`startup.rdata`':** (comma-separated values) Controls whether an existing '`./.RData`' file should be processed or not. If `"remove"`, it will be skipped by automatically removing it. If `"rename"`, it will be renamed to '`./.RData.YYYYMMDD_hhmmss`' where the timestamp is the last time the file was modified. If `"prompt"`, the user is prompted whether they want to load the file or rename it. In non-interactive session, `"prompt"` will fallback to loading the content (default). To fallback to renaming the file, use `"prompt,rename"`. Note that in contrast to R and `R CMD BATCH file.R`, `Rscript` does *not* load '`.RData`' files unless command-line option `--restore` is specified. (Default: not specified)

Controls what validation checks are performed at startup

`R_STARTUP_CHECK` / **'`startup.check`':** (logical) Controls the default value of argument `check` of `startup()`. (Default: `TRUE`)

`R_STARTUP_CHECK_OPTIONS_IGNORE` / **'`startup.check.options.ignore`':** (character vector or comma-separated character string) Names of R options that should *not* be validated at the end of the `startup()` process. (Default: `"error"`)

Settings useful for debugging and prototyping

`R_STARTUP_DEBUG` / **'`startup.debug`':** (logical) Controls the default value of argument `debug` of `startup()`. (Default: `FALSE`)

'`startup.commandArgs`': (character vector) Overrides the command-line arguments that `startup()` uses, which can be useful to prototype and test alternative ways that R might be launched. (Default: `base::commandArgs()`)

`R_STARTUP_TIME` / **'`startup.time`':** (POSIX timestamp; character string) Overrides the current timestamp, which can be useful to prototype and test functionalities that depend on the current time, e.g. inclusion and exclusion of files based on `when=<periodicity>` tags. The specified string must be parsable by `base::as.POSIXct()`. (Default: not specified)

`startup_session_options` *Record R session information as options*

Description

Record R session information as options

Usage

```
startup_session_options(action = c("update", "overwrite", "erase"))
```

Arguments

`action`	If `"update"` or `"overwrite"`, R options `"startup.session.*"` are set. If `"update"`, then such options that are not already set are updated. If `"erase"`, any existing `"startup.session.*"` options are removed.

Value

Returns invisibly a named list of the options prefixed `"startup.session."`:

`startup.session.startdir` (character) the working directory when the **startup** was first loaded. If `startup::startup()` is called at the very beginning of the '`.Rprofile`' file, this is also the directory that the current R session was launched from.

`startup.session.starttime` (POSIXct) the time when the **startup** was first loaded.

`startup.session.id` (character) a unique ID for the current R session.

`startup.session.dumpto` (character) a session-specific name that can be used for argument `dumpto` of dump.frames() (also for dumping to file).

Examples

```
opts <- startup::startup_session_options()
opts
```

sysinfo *Information on the current R session*

Description

Information on the current R session

Usage

```
sysinfo()
```

Value

A named list.

Examples

```
startup::sysinfo()
```

warn *Produce a warning*

Description

Produce a warning

Usage

```
warn(..., call. = FALSE, immediate. = TRUE, domain = NULL)
```

Arguments

`..., domain`	The message and optionally the domain used for translation. The ... arguments are passed to base::sprintf to create the message string.
`call.`	If base::TRUE, the call is included in the warning message, otherwise not.
`immediate.`	If base::TRUE, the warning is outputted immediately, otherwise not.

Index