Package 'prrd'

November 7, 2026

Type Package

Title Parallel Runs of Reverse Depends

Version 0.1.0

Date 2026-11-07

Description Reverse depends for a given package are queued such that multiple workers can run the reverse-dependency tests in parallel where operations are robust to concurrent workers writing. Work queue management is provided by package 'liteq' from which an approach for dealing with concurrent 'sqlite' writes is also borrowed.

License GPL (>= 2)

Imports config, liteq, data.table, crayon, DBI, RSQLite

Suggests docopt, foghorn, anytime

URL https://github.com/eddelbuettel/prrd, https://dirk.eddelbuettel.com/code/prrd.html

BugReports https://github.com/eddelbuettel/prrd/issues

RoxygenNote 6.0.1

Encoding UTF-8

NeedsCompilation no

Author Dirk Eddelbuettel [aut, cre] (ORCID: <https://orcid.org/0000-0001-6419-907X>)

Maintainer Dirk Eddelbuettel <edd@debian.org>

Repository CRAN

Date/Publication 2026-11-07 16:30:07 UTC

Contents

prrd-package 2
dequeueJobs 2
enqueueJobs 3

getDatabaseConnection 4
runSanityChecks 5
summariseQueue 7

Index **8**

`prrd-package` *Parallel Runs of Reverse Depends*

Description

Reverse depends for a given package are queued such that multiple workers can run the reverse-dependency tests in parallel where operations are robust to concurrent workers writing. Work queue management is provided by package 'liteq' from which an approach for dealing with concurrent 'sqlite' writes is also borrowed.

Package Content

Index of help topics:

`dequeueJobs`	`Dequeue and run reverse-dependency checks, possibly in par`
`enqueueJobs`	`Enqueues reverse-dependent packages`
`getDatabaseConnection`	`Database Helper Functions`
`prrd-package`	`Parallel Runs of Reverse Depends`
`runSanityChecks`	`Various Helper Functions`
`summariseQueue`	`Summarisse results from a reverse-dependency check`

Maintainer

Dirk Eddelbuettel <edd@debian.org>

Author(s)

Dirk Eddelbuettel [aut, cre] (ORCID: <https://orcid.org/0000-0001-6419-907X>)

`dequeueJobs` *Dequeue and run reverse-dependency checks, possibly in parallel*

Description

Dequeue package for (potentially parallel) reverse-dependency check

Usage

```
dequeueJobs(package, directory, exclude = NULL, date = format(Sys.Date()))

dequeueDepends(package, directory)
```

Arguments

package	A character variable denoting a package
directory	A character variable denoting a directory for the queuefile
exclude	An optional character variable denoting an exclusion set csv file.
date	Optional character variable describing a date (as part of the queue file, default is current date.

Details

This function consumes previously enqueued jobs for reverse dependency checks. It is set up in such a way that multiple distinct and independent process can run checks in parallel without effecting each other. If the underlying queue file is on a network drive, this should may also work across multiple machines.

Value

A queue is create as a side effect, its elements are returned invisibly in verbose mode only.

Author(s)

Dirk Eddelbuettel

---

enqueueJobs *Enqueues reverse-dependent packages*

Description

Enqueue package for subsequent reverse-dependency check

Usage

```
enqueueJobs(package, directory, dbfile = "", addfailed = FALSE)

enqueueDepends(package, directory)
```

Arguments

package	A character variable denoting a package
directory	A character variable denoting a directory
dbfile	Optional character with path to previous runs db file
addfailed	Optional logical swith to add previous failures

Details

This function relies on the `available.packages` function from R along with the `liteq` package. The `getQueueFile` function is used to determine the queue file directory and name.

Value

A queue is create as a side effect, its elements are returned invisibly

Author(s)

Dirk Eddelbuettel

Examples

```
## Not run:
td <- tempdir()
options(repos=c(CRAN="https://cloud.r-project.org"))
jobsdf <- enqueueJobs(package="digest", directory=td)

## End(Not run)
```

---

getDatabaseConnection *Database Helper Functions*

Description

Several Database Connection Helper Functions

Usage

```
getDatabaseConnection(file)

createTable(con)

insertRow(con, df)

createRunDataTable(con)

dbWithWriteTransaction(con, code, retry = TRUE)

dbExecuteLocked(file, code, ...)

dbGetQueryWrapped(file, code)

createTableLocked(file)

insertRowLocked(file, df)

createRunDataTableLocked(file)

dbWriteTableLocked(file, table, df, append = FALSE, overwrite = TRUE)
```

Arguments

file	A character variable pointing to a sqlite3 file
con	A database connection object
df	A one-row data.frame with results to be appended
code	Exression to be executed
retry	Boolean value, defaults to 'TRUE'
...	Additional parameters, currently unused
table	Character value with the database table to write to
append	Boolean value whether to append to table or not, depends to 'FALSE'
overwrite	Boolean value whether to overwrite table or not, depends to 'TRUE'

Details

These functions return a connection, create a table and insert a row of data, respectively.

Value

A connection object

Author(s)

Dirk Eddelbuettel

---

runSanityChecks *Various Helper Functions*

Description

These functions create a data directory name given a package name and additional optional arguments which can be used to store test meta-data and results, return a queue file name, return the configuration object (retrieved from several standard locations) and load the exclusion set.

Usage

```
runSanityChecks()

runEnqueueSanityChecks()

getDataDirectory(package, date = format(Sys.Date()),
  path = getOption("prrd.queue_directory", "."), sep = "_")

getQueueFile(package, date = format(Sys.Date()),
  path = getOption("prrd.queue_directory", "."), sep = "_")

getConfig(debug = FALSE)

getExclusionSet(exclusionfile)
```

Arguments

package	Character variable providing the package name.
date	Optional character variable describing a date, default is current date.
path	Optional path, default is option value or current directory.
sep	Optional character with component separator, default is "_".
debug	Optional boolean
exclusionfile	Character variable with path and file for an optional exclusion set.

Details

Additional helper functions get particular directories, config settings or check some assumptions about files in PATH and alike,

Value

For getDataDirectory, a directory name. For getQueueFile, a queue database file name (used only internally). For getConfig, a named vector with config entries. For getExclusionSet a character vector of package names. Nothing is returned by runSanityChecks which is invoked for its side effect of validation (currently only if run-xvfb-safe is in the PATH).

Config File

An optional config file can be used to set several configuration variables. Currently supported are

setup The path to an R file that will be source'ed.

workdir The directory used for the parallel run of reverse depends. May be an absolute or relative path.

libdir The directory passed to .libPaths allow for additional build-dependencies. May be an absolute or relative path.

debug A boolean switch to enable more debugging output.

verbose A boolean switch to enable more verbose output.

The setup field is particularly useful as the file being sourced can contain other R expressions setting environment variables or options. It is (currently) searched in the four locations ".prrd.yaml", "~/.R/prrd.yaml", "~/.prrd.yaml", and "/etc/R/prrd.yaml". The first file found ends the search and is used, content is not merged from several files.

Author(s)

Dirk Eddelbuettel

summariseQueue *Summarisse results from a reverse-dependency check*

Description

Summarise results from (potentially parallel) reverse-dependency check

Usage

```
summariseQueue(package, directory, dbfile = "", extended = FALSE,
  foghorn = FALSE)
```

Arguments

package	A character variable denoting a package
directory	A character variable denoting a directory
dbfile	A character variable for an optional override which, if present, is used over 'package' and 'directory'.
extended	A boolean variable to select extended analysis of failures, default is FALSE which skips this.
foghorn	A boolean variable to invoke the **foghorn** to retrieve and review CRAN result status, default is FALSE which skips this.

Value

NULL, invisibly

Author(s)

Dirk Eddelbuettel

Index

∗ **package**
    prrd-package, 2

available.packages, *3*

createRunDataTable
        *(*getDatabaseConnection*)*, 4
createRunDataTableLocked
        *(*getDatabaseConnection*)*, 4
createTable *(*getDatabaseConnection*)*, 4
createTableLocked
        *(*getDatabaseConnection*)*, 4

dbExecuteLocked
        *(*getDatabaseConnection*)*, 4
dbGetQueryWrapped
        *(*getDatabaseConnection*)*, 4
dbWithWriteTransaction
        *(*getDatabaseConnection*)*, 4
dbWriteTableLocked
        *(*getDatabaseConnection*)*, 4
dequeueDepends *(*dequeueJobs*)*, 2
dequeueJobs, 2

enqueueDepends *(*enqueueJobs*)*, 3
enqueueJobs, 3

getConfig *(*runSanityChecks*)*, 5
getDatabaseConnection, 4
getDataDirectory *(*runSanityChecks*)*, 5
getExclusionSet *(*runSanityChecks*)*, 5
getQueueFile *(*runSanityChecks*)*, 5

insertRow *(*getDatabaseConnection*)*, 4
insertRowLocked
        *(*getDatabaseConnection*)*, 4

prrd *(*prrd-package*)*, 2
prrd-package, 2

runEnqueueSanityChecks
        *(*runSanityChecks*)*, 5
runSanityChecks, 5

summariseQueue, 7