Package 'drat'

April 13, 2022

Type Package

Title 'Drat' R Archive Template

Version 0.2.3

Date 2022-04-13

Author Dirk Eddelbuettel with contributions by Carl Boettiger, Neal Fultz, Sebastian Gibb, Colin Gillespie, Jan Górecki, Matt Jones, Thomas Leeper, Steven Pav, Jan Schulz, Christoph Stepper, Felix G.M. Ernst and Patrick Schratz.

Maintainer Dirk Eddelbuettel <edd@debian.org>

Depends R (>= 3.6.0)

Imports utils

Suggests git2r, simplermarkdown

VignetteBuilder simplermarkdown

Description Creation and use of R Repositories via helper functions to insert packages into a repository, and to add repository information to the current R session. Two primary types of repositories are support: gh-pages at GitHub, as well as local repositories on either the same machine or a local network. Drat is a recursive acronym: Drat R Archive Template.

License GPL (>= 2)

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

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

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-13 13:32:29 UTC

R topics documented:

`drat-package` *Easy-to-use package repository creation and access*

Description

The drat package permits user to create and use ad-hoc package repositories. It takes advantage of GitHub accounts and 'gh-pages' branches which automatically become web-accessible and can be used to provide a repository. Alternatively, custom repository paths and addresses can be used.

Details

Given a user account on GitHub, say, 'eddelbuettel', and a repository 'drat', we can infer an top-level repostory URL as such as `https://eddelbuettel.github.io/drat/` by supplying only the username (as the rest is inferred by defaults). This allows us to create easily useable, identifiable and shareable per-user repositories—without the user having to create and administer a webserver anywhere.

Two higher level functions then allow both insertion of (source or binary) packages, as well as addition of a given drat repository to an R session so that package in the repository can be accesses.

Author(s)

Dirk Eddelbuettel

Maintainer: Dirk Eddelbuettel <edd@debian.org>

References

The R Installation and Administration manual has more and details about repository creation

See Also

`update.packages`, `available.packages`, `install.packages`

Examples

```
## Not run:
drat::addRepo("eddelbuettel")   # adds the repo of GitHub user 'eddelbuettel'

## End(Not run)
```

---

addRepo *Add a (drat) repository to the current session*

Description

R can use multiple archives: CRAN, BioConductor and Omegahat have been supported for years. It is equally easy to add local archives from the same machine, or local network, or university / company network as well as other publically available repositories. This function aids in the process, and defaults to adding a 'drat' archive at GitHub under the given account.

Usage

```
addRepo(account, alturl)

add(...)
```

Arguments

account	Character vector with one or more GitHub account for which a 'drat' archive is to be added.
alturl	Alternative repo specification with a complete url string. If 'alturl' is provided, a single 'account' must be provided as well. For file-based access, the URL format has to follow the `file:/some/path/` format starting with 'file' followed by a single colon.
...	For the aliases variant, a catch-all collection of parameters.

Details

This function retrieves the current set of repositories (see `getOption("repos")` for the current values) and adds (or overwrites) the entry for the given 'account'. For non-GitHub repositories an alternative URL can be specified as 'alturl' (and assigned to 'account' as well).

An aliased function `add` is also available, but not exported via `NAMESPACE` to not clobber a possibly unrelated function; use it via `drat:::add()`.

Value

The altered set of repositories

Author(s)

Dirk Eddelbuettel

Examples

```
## Not run:
  addRepo("drat")                           # adds GitHub repo via default URL
  addRepo(c("eddelbuettel", "ghrr"))        # ditto but adds two repos at once

  addRepo("LocalRepo", "file:/nas/R/repo")  # adds local file-based repo,
                                            # assumes you can read /nas/R/repo

## End(Not run)
```

`archivePackages` *Move older copies of packages to an archive*

Description

The function moves older versions of packages into a CRAN-style archive folder.

Usage

```
archivePackages(
  repopath = getOption("dratRepo", "~/git/drat"),
  type = c("source", "binary", "mac.binary", "mac.binary.el-capitan",
    "mac.binary.mavericks", "win.binary", "both"),
  pkg,
  version = getRversion()
)

archivePackagesForAllRversions(
  repopath = getOption("dratRepo", "~/git/drat"),
  type = c("source", "binary", "mac.binary", "mac.binary.el-capitan",
    "mac.binary.mavericks", "win.binary", "both"),
  pkg
)
```

Arguments

`repopath`	Character variable with the path to the repo; defaults to the value of the "dratRepo" option with ""~/git/drat"" as fallback
`type`	Character variable for the type of repository, so far "source", "binary", "win.binary", "mac.binary", "mac.binary.mavericks", "mac.binary.el-capitan" or "both"
`pkg`	Optional character variable specifying a package name(s), whose older versions should be archived. If missing (the default), archiving is performed on all packages.
`version`	R version information in the format `X.Y` or `X.Y.Z`. Only used, if archiving binary packages. (default: `version = getRversion()`). If `version = NA`, all available R versions will be used. If `version = NULL`, this defaults to `getRversion()`.

Details

This function is still undergoing development and polish and may change in subsequent versions.

Author(s)

Thomas J. Leeper

Examples

```
## Not run:
  archivePackages()   # archive all older package versions
  archivePackages(pkg = "drat")  # archive older copies of just one package

## End(Not run)
```

---

getPackageInfo *Get information from a binary package*

Description

This function returns the compile-time information added to the `DESCRIPTION` file in the package.

Usage

```
getPackageInfo(file)
```

Arguments

`file`	the fully qualified path of the package

Value

A named vector with several components

Note

This is an internal function, use `:::` to access it from outside the internal package code.

Author(s)

Dirk Eddelbuettel

identifyPackageType *Identifies the package type from a filename*

Description

This function identifies the package type from a filename.

Usage

```
identifyPackageType(file, pkginfo = getPackageInfo(file))
```

Arguments

file	An R package in source or binary format,
pkginfo	information on the R package referenced by file

Details

The returned string is suitable for write_PACKAGES().

Value

string Type of the supplied package.

Note

This is an internal function, use ::: to access it from outside the internal package code.

Author(s)

Jan Schulz and Dirk Eddelbuettel

initRepo *Intialize a git repo for drat*

Description

This helper function creates a new repository, creates and checks out the default GitHub Pages location (either the 'gh-pages' branch or directory 'docs') and fills it with the required new paths.

Usage

```
initRepo(
  name = "drat",
  basepath = getOption("dratDirectory", "~/git"),
  location = getOption("dratBranch", "gh-pages")
)
```

Arguments

name	A character variable with the name the new repository, the default is "drat".
basepath	A character variable with path to the directory in which the new repository is to be created. The default value is "~/git" and can be overriden via option 'dratDirectory'.
location	A character variable with the GitHub Pages location: either "gh-pages" indicating a branch of that name, or "docs/" directory in the main branch. The default value can be overridden via the "dratBranch" option.

Details

Currently only 'src/contrib' for source repositories is supported by this function. The `insertPackage()` function knows to deal with binaries for different architectures.

The function also installs a top-level `index.html` file to ensure external tests against the repository (as for example done by CRAN if you list the repository as an 'Additional_repositories' in a package) do not return a '404' error.

Value

The function is invoked for its side-effects and only returns `NULL` invisibly.

Author(s)

Dirk Eddelbuettel

`insertPackage` *Insert a package source or binary file into a drat repository*

Description

R can use multiple archives: CRAN, BioConductor and Omegahat have been supported for years. It is equally easy to add local archives from the same machine, or local network, or university / company network as well as other publically available repositories. This function aids in the process, and defaults to inserting a given source archive into a given repository.

Usage

```
insertPackage(
  file,
  repodir = getOption("dratRepo", "~/git/drat"),
  commit = FALSE,
  pullfirst = FALSE,
  action = c("none", "archive", "prune"),
  location = getOption("dratBranch", "gh-pages"),
  ...
)
```

```
insertPackages(file, ...)

insert(...)
```

Arguments

file	One or more R package(s) in source or binary format
repodir	A local directory corresponding to the repository top-level directory.
commit	Either boolean toggle to select automatic git operations 'add', 'commit', and 'push' or, alternatively, a character variable can be used to specify a commit message; this also implies the 'TRUE' values in other contexts.
pullfirst	Boolean toggle to call `git pull` before inserting the package.
action	A character string containing one of: "none" (the default; add the new package into the repo, effectively masking previous versions), "archive" (place any previous versions into a package-specific archive folder, creating such an archive if it does not already exist), or "prune" (calling pruneRepo).
location	A character variable with the GitHub Pages location: either "gh-pages" indicating a branch of that name, or "docs/" directory in the main branch. The default value can be overridden via the "dratBranch" option.
...	For `insert` the aliases variant, a catch-all collection of parameters. For `insertPackage` arguments passed to `write_PACKAGES` currently include `latestOnly`, for which the default value is set here to `FALSE`. See write_PACKAGES.

Details

This function inserts the given (source or binary) package file into the given (local) package repository and updates the index. By setting the `commit` option to `TRUE`, one can then push to a remote git code repository. If the git2r package is installed, it is used for the interaction with the git repository; otherwise the `git` shell command is used.

An aliased function `insert` is also available, but not exported via `NAMESPACE` to not clobber a possibly unrelated function; use it via `drat:::insert()`.

The function also checks for a top-level `index.html` file to ensure external tests against the repository (as for example done by CRAN if you list the repository as an 'Additional_repositories' in a package) do not return a '404' error. If missing, a simple one-line example is shown.

Value

NULL is returned.

Options

Set using options

`dratRepo` Path to git repo. Defaults to `~/git/drat`

`dratBranch` The git branch to store packages on. Defaults to `gh-pages`

Author(s)

Dirk Eddelbuettel

Examples

```
## Not run:
  insertPackage("foo_0.2.3.tar.gz")   # inserts into (default) repo
  insertPackage("foo_0.2.3.tar.gz", "/nas/R/")  # ... into local dir

## End(Not run)
## Not run:
  insertPackage("foo_0.2.3.tar.gz", action = "prune")   # prunes any older copies
  insertPackage("foo_0.2.3.tar.gz", action = "archive")   # archives any older copies

## End(Not run)
```

---

pruneRepo *Prune repository from older copies of packages*

Description

The function determines which packages in a repositories can be removed as they are being 'shadowed' by a newer version of the same packages.

Usage

```
getRepoInfo(
  repopath = getOption("dratRepo", "~/git/drat"),
  type = c("source", "binary", "mac.binary", "mac.binary.el-capitan",
    "mac.binary.mavericks", "win.binary", "both"),
  pkg,
  version = getRversion(),
  location = getOption("dratBranch", "gh-pages")
)

pruneRepo(
  repopath = getOption("dratRepo", "~/git/drat"),
  type = c("source", "binary", "mac.binary", "mac.binary.el-capitan",
    "mac.binary.mavericks", "win.binary", "both"),
  pkg,
  version = getRversion(),
  remove = FALSE,
  location = getOption("dratBranch", "gh-pages")
)

pruneRepoForAllRversions(
  repopath = getOption("dratRepo", "~/git/drat"),
```

```
  type = c("source", "mac.binary", "mac.binary.el-capitan", "mac.binary.mavericks",
    "win.binary", "both"),
  pkg,
  remove = FALSE
)

updateRepo(
  repopath = getOption("dratRepo", "~/git/drat"),
  type = c("source", "mac.binary", "mac.binary.el-capitan", "mac.binary.mavericks",
    "win.binary", "both"),
  version = NA,
  ...
)
```

Arguments

repopath	Character variable with the path to the repo; defaults to the value of the "dratRepo" option with ""~/git/drat"" as fallback
type	Character variable for the type of repository, so far "source", "binary", "win.binary", "mac.binary", "mac.binary.mavericks", "mac.binary.el-capitan" or "both"
pkg	Optional character variable specifying a package name, whose older versions should be pruned. If missing (the default), pruning is performed on all packages.
version	R version information in the format `X.Y` or `X.Y.Z`. Only used, if pruning binary packages. (default: `version = getRversion()`). If `version = NA`, all available R versions will be used. If `version = NULL`, this defaults to `getRversion()`.
location	An optional character variable with the GitHub Pages location: either "gh-pages" indicating a branch of that name, or "docs/" directory in the main branch. The default value can be overridden via the "dratBranch" option.
remove	Character or logical variable indicating whether files should be removed. Nothing happens if 'FALSE'. If different from (logical) 'FALSE' and equal to character "git" files are removed via `git rm` else via a straight file deletion.
...	For `updateRepo` a catch-all collection of parameters. Arguments passed to `update_PACKAGES` currently include `latestOnly`, for which the default value is set here to `FALSE`. See `update_PACKAGES`. Please note that this has an effect for `update_PACKAGES` only, if new packages are found, e.g. manually added.

Details

Given a package name, R will always find the newest version of that package. Older versions are therefore effectively shadowed and can be removed without functionally changing a repository.

However, if a current package file is removed without `pruneRepo`, the PACKAGES, PACKAGES.gz and PACKAGES.rds file might be not up to date. To ensure the correct information is available in these indices, run `updateRepo`.

These functions are still undergoing development and polish and may change in subsequent versions.

Value

A data frame describing the repository is returned containing columns with columns “file”, “package” (just the name), “version” and a logical variable “newest” indicating if the package can be removed.

Author(s)

Dirk Eddelbuettel

Index