Package 'GRANBase'

May 9, 2018

Type Package

Title Creating Continuously Integrated Package Repositories from Manifests

Version 2.5.0

Author Gabriel Becker[aut,cre], Cory Barr [cre,ctb], Dinakar Kulkarni [aut,ctb]

Maintainer Gabriel Becker <becker.gabriel@gene.com>

Copyright Genentech Inc

Description Repository based tools for department and analysis level reproducibility. 'GRANBase' allows creation of custom branched, continuous integration-ready R repositories, including incremental testing of only packages which have changed versions since the last repository build.

License Artistic-2.0

Depends GRANCore, switchr (>= 0.9.28), methods

Imports tools, utils, htmlTable (>= 1.11.0), dplyr, sendmailR, covr, RCurl, jsonlite, stringi, stats, markdown

Suggests parallel, rmarkdown, hexSticker, knitr

VignetteBuilder knitr

SystemRequirements svn, git

URL https://github.com/gmbecker/gRAN

BugReports https://github.com/gmbecker/gRAN/issues

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-05-09 17:11:05 UTC

R topics documented:

`addPkg,GRANRepository-method`
addPkg

Description

Add a package to the manifest for a GRANRepository

Usage

```
## S4 method for signature 'GRANRepository'
addPkg(x, ..., rows = makeManifest(...),
  versions = data.frame(name = manifest_df(rows)$name, version =
  NA_character_, stringsAsFactors = FALSE), replace = FALSE)
```

Arguments

x	A GRANRepository object
...	passed to manifest method for addPkg
rows	data.frame or unspecified. passed to manifest method for addPkg
versions	data.frame passed to manifest method for addPkg
replace	logical. Should the information in .../rows replace existing rows for the same pacakge? Defaults to FALSE, in which case an error is thrown.

Value

x with the specified package(s) added to the associated manifest

Examples

```
man = GithubManifest("gmbecker/switchr")
repo = GRANRepository(man, basedir = tempdir())
repo = addPkg(repo, rows = GithubManifest("gmbecker/rpath"))
```

---

buildBadge *Create Badges for build status*

Description

Create Badges for build status

Usage

```
buildBadge(status, pkg_name)
```

Arguments

status	The build status for the package
pkg_name	Name of the package

Value

Badge href tag

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

buildBranchesInRepo *Build SCM Checkouts Into Repository Directory*

Description

Create the tarballs in the new repo from the SCM branch locs

Usage

```
buildBranchesInRepo(repo, cores = 1, temp = FALSE, incremental = TRUE,
  manifest = manifest_df(repo))
```

Arguments

repo	a GRANRepository object
cores	number of cores to use during build process. defaults to (parallel:::detectCores() - 1)
temp	logical. whether we are building the temp or final version of the repository
incremental	logical. whether packages should only be rebuilt if the version number has increased. Default is TRUE
manifest	data.frame containing a GRAN manifest of pkgs to build. Defaults to the full manifest associated with repo

Value

a list with success and fail elements containing the directories which succeeded and failed the build

Author(s)

Cory Barr, Gabriel Becker

buildReport *buildReport*

Description

Create a build report for a repository reflecting the latest build

Usage

```
buildReport(repo, theme = "bootstrap",
  reportfile = file.path(destination(repo), "buildreport.html"),
  riskrpt = FALSE, jsonrpt = TRUE, splashname = "index.html")
```

Arguments

repo	A GRANRepository object
theme	CSS+JS theme. bootstrap, foundation, semanticui or jqueryui
reportfile	File path of the HTML report
riskrpt	Whether to build the risk report
jsonrpt	Whether to create a JSON version of the build report
splashname	Filename for the package HTML splash page

Value

None

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

---

buildReportURL *Constructs the gRAN build report URL*

Description

Constructs the gRAN build report URL

Usage

```
buildReportURL(repo)
```

Arguments

repo	A gRAN repo object

Note

This function is not intended for direct use by the end user.

`buildRiskReport` *Build risk-assessment for proposed package updates*

Description

buildRiskReport

Usage

```
buildRiskReport(repo, to_update = old.packages(repos = repo_urls),
  important_pkgs = installed.packages(lib.loc = liblocs)[, "Package"],
  liblocs = .libPaths(), repo_urls = getOption("repos"),
  report_file = file.path(destination(repo), "update-risk.html"),
  theme = "bootstrap")
```

Arguments

`repo`	The name of a GRAN repository to use. Assumes that a a package named GRAN<repo> is available to load.
`to_update`	vector of package names which may be updated, or a matrix output from `old.packages`. Defaults to all packages which are out of date
`important_pkgs`	list of packages to check for risk of change cascades from updating the packages in `to_update`. Defaults to all installed packages
`liblocs`	the library locations to look for installed packages
`repo_urls`	The repositories to check for new versions of packages
`report_file`	File where HTML report will be written to
`theme`	CSS theme. bootstrap, foundation, semanticui or jqueryui

Details

Generates an HTML report identifying which packages have updates available, and which of the specified important packages may be effected by installing those new versions.

Value

none. Writes HTML report with risk assessment

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

`clear_temp_fils` *Clear packages and temporary files from repo build process*

Description

These are convenience functions which clears the intermediate files generated during the build process. This is important when, e.g., building a repository for the first time with a new version of R.

Usage

```
clear_temp_fils(repo, checkout = FALSE)

clear_repo(repo, all = TRUE, checkout = FALSE, archivedir = NA)
```

Arguments

`repo`	GRANRepository - The repository to clean
`checkout`	logical - Should the checkouts of packages also be cleared. Generally this is not necessary (default: FALSE)
`all`	logical - Should temporary artifacts from the build process also be cleared (via automatically calling clear_temp_fils). Defaults to TRUE
`archivedir`	character - Optional. A directory where build packages deployed to the repository will be archived. Package versions already in the archive will not be overwritten.

Details

`clear_repo` removes packages deployed into the destination repository, updates the PACKAGES and PACKAGES.gz files, and resets the build results within the GRANRepository object. `clear_temp_fils` clears intermediate files from the library location used during building, the temporary repository, the package staging area, and the store of install- and check-results.

Value

The GRANRepository object, ready to be rebuilt.

Author(s)

Gabriel Becker

createHyperlink *Creates a href tag*

Description

Creates a href tag

Usage

```
createHyperlink(string, label = "")
```

Arguments

string	URL string
label	Label for href

Value

href tag

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

createJSON *Create JSON representation of package information*

Description

Create JSON representation of package information

Usage

```
createJSON(repo, pkg_name, descr_df, scm_df, docdir, rev_deps,
  suffix = paste0("_", descr_df$Version, ".json"))
```

Arguments

repo	A GRAN repo object
pkg_name	Name of the GRAN package
descr_df	data.frame representation of DESCRIPTION file
scm_df	data.frame representation of GRAN manifest object
docdir	Directory where the JSON doc will be written
rev_deps	data.frame representing pkg_name's reverse deps
suffix	Suffix for the JSON file

Value

None. Write JSON file to disk

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

See Also

`manifest_df` for generating scm_df and `generateDescInfo` for generating descr_df.

`createMeta` *Create package metadata files in the "Meta" folder*

Description

Create package metadata files

Usage

```
createMeta(repo, repodest = destination(repo), metadir = metadatadir(repo),
  archive_dir = archivedir(repo), serialize_json = FALSE)
```

Arguments

`repo`	A GRAN repo object
`repodest`	The repo destination (something that looks like BASE_REPO_DIR/src/contrib)
`metadir`	The directory containing metadata files
`archive_dir`	Directory containing package archive
`serialize_json`	Serialize the RDS metadata files as JSON

Author(s)

Dinakar Kulkarni

createSticker *Create hex stickers for packages*

Description

Create hex stickers for packages

Usage

```
createSticker(pkgName, destination)
```

Arguments

pkgName	Name of the package
destination	Directory where the sticker will be saved

Value

None

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

deltaDF *Returns the difference between 2 data frames*

Description

Returns the difference between 2 data frames

Usage

```
deltaDF(new_df, old_df)
```

Arguments

new_df	The new dataframe which you want to compare
old_df	An older dataframe of the same structure

Value

Differences as a dataframe of the same structure

Note

This function is not intended for direct use by the end user.

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

See Also

anti_join

`determinePkgURL` *Generate package's external URL after validation info as data.frame*

Description

Generate package's external URL after validation info as data.frame

Usage

```
determinePkgURL(pkg_name)
```

Arguments

`pkg_name`	The name of the package (string)

Value

Package external URL

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

emailNotifier *Send email notifications to maintainers whose builds failed*

Description

Send email notifications to maintainers whose builds failed

Usage

```
emailNotifier(repo, mailopts = email_options(repo), attachments = NULL)
```

Arguments

repo	A gRAN repo object
mailopts	Email options as a list
attachments	Files with full paths, as an array

Value

None

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

See Also

getFailureInfo for creating failed pkg manifests, sendMail for sending emails

emailTag *Create a mailto tag for email IDs*

Description

Create a mailto tag for email IDs

Usage

```
emailTag(item)
```

Arguments

item	A string with the email ID

Value

href tag

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

`encode_string` *Convert string to numeric representation*

Description

Convert string to numeric representation

Usage

```
encode_string(x)
```

Arguments

`x`	String

Value

Numeric representation of string

Note

This function is not intended for direct use by the end user.

`extPkgURL` *Wrapper for determinePkgURL*

Description

Wrapper for determinePkgURL

Usage

```
extPkgURL(desc_field, as_string = TRUE)
```

Arguments

`desc_field`	The DESCRIPTION field which has mutiple package names
`as_string`	Return as string

Value

Package external URL

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

`extPkgURL_old` *Wrapper for determinePkgURL. Deprecate due to performance issues.*

Description

Wrapper for determinePkgURL. Deprecate due to performance issues.

Usage

```
extPkgURL_old(desc_field, as_string = TRUE)
```

Arguments

`desc_field`	The DESCRIPTION field which has mutiple package names
`as_string`	Return as string

Value

Package external URL

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

`generateDescInfo` *Converts a DESCRIPTION file to a data.frame*

Description

Converts a DESCRIPTION file to a data.frame

Usage

```
generateDescInfo(pkg_path, encoding = "")
```

Arguments

pkg_path	The path preceding the location of the DESCRIPTION file
encoding	If there is an Encoding field, to what encoding should re-encoding be attempted? The other values are as used by iconv, so the default "" indicates the encoding of the current locale

Value

If a DESCRIPTION file for the given package is found and can successfully be read, this function returns a data.frame ontaining of the fields as headers and the tags as rows

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

---

getFailureInfo *Creates a dataframe containing information regarding packages that had a failed status*

Description

Creates a dataframe containing information regarding packages that had a failed status

Usage

```
getFailureInfo(repo)
```

Arguments

repo	A GRAN repo object

Value

Dataframe containing package info that failed

Note

This function is not intended for direct use by the end user.

See Also

repo_results for repo results as a dataframe

getOS *Get the OS Type*

Description

Get the OS Type

Usage

```
getOS()
```

Value

OS Type

Note

This function is not intended for direct use by the end user.

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

identifyRisk *identifyRisk*

Description

Identify packages which could possibly be effected by updating the specified list of packages to their latest versions.

Usage

```
identifyRisk(repo, to_update = old.packages(repos = repo_urls),
  liblocs = .libPaths(), important_pkgs = installed.packages(lib.loc =
  liblocs)[, "Package"], repo_urls = getOption("repos"))
```

Arguments

repo	The name of a GRAN repository to use. Assumes that a a package named GRAN<repo> is available to load.
to_update	vector of package names which may be updated, or a matrix output from old.packages. Defaults to all packages which are out of date
liblocs	the library locations to look for installed packages
important_pkgs	list of packages to check for risk of change cascades from updating the packages in to_update. Defaults to all installed packages
repo_urls	The repositories to check for new versions of packages

Value

A list containing two named lists: splash_damage and in_danger. splash_damage lists the packages potentially affected by updating each package in `to_update`. in_danger lists the packages from `to_update` that affect each package in `important_pkgs` (packages which are unaffected are omitted).

Author(s)

Gabriel Becker

`isValidEmail` *Checks whether an email ID is valid*

Description

Checks whether an email ID is valid

Usage

```
isValidEmail(email_id)
```

Arguments

`email_id`	Email ID as a string

Value

Boolean

Note

This function is not intended for direct use by the end user.

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

makeRepo *makeRepo*

Description

Make a package repository containing a specified set of packages from various sources

Usage

```
makeRepo(x, cores = 1, build_pkgs = NULL, scm_auth = list(bioconductor.org
  = c("readonly", "readonly")), ...)

## S4 method for signature 'PkgManifest'
makeRepo(x, cores = 1, build_pkgs = NULL,
  scm_auth = list(bioconductor.org = c("readonly", "readonly")), ...)

## S4 method for signature 'SessionManifest'
makeRepo(x, cores = 1, build_pkgs = NULL,
  scm_auth = list(bioconductor.org = c("readonly", "readonly")), ...)

## S4 method for signature 'GRANRepository'
makeRepo(x, cores = 1, build_pkgs = NULL,
  scm_auth = list(bioconductor.org = c("readonly", "readonly")), ...)

## S4 method for signature 'character'
makeRepo(x, cores = 1, build_pkgs = NULL,
  scm_auth = list(bioconductor.org = c("readonly", "readonly")), ...)
```

Arguments

x	The object containing the information necessary to create the repository
cores	The number of cores on the local machine to use during building
build_pkgs	The names of the packages to (re) build and test within the repository. Defaults to NULL which builds all packages in the manifest
scm_auth	A named list containing the information necessary to check out package sources. The list elements (assumed to be a character vector of length 2, user then password) are applied when the name is contained in a package's url
...	Additional arguments, typically used for the construction of a GRANRepository object if one does not already exist.

Value

A GRANRepository object which has used to create a repository.

References

Becker G, Barr C, Gentleman R, Lawrence M; Enhancing Reproducibility and Collaboration via Management of R Package Cohorts. Journal of Statistical Software, 81(1). 2017. doi: 10.18637/jss.v082.i01

`makeWinBins` *Make Windows binary packages*

Description

Create Windows binary builds (only works on Windows machines)

Usage

```
makeWinBins(repo, cores = 1,
  virtualstore = file.path(Sys.getenv("LOCALAPPDATA"), "VirtualStore"))
```

Arguments

`repo`	A GRANRepository object
`cores`	Number of cores to use during build process
`virtualstore`	Windows VM directory where built binaries are stored

Value

None

Author(s)

Dinakar Kulkarni

`manifestReport` *manifestReport*

Description

Build a package manifest report for a GRAN repository

Usage

```
manifestReport(repo, theme = "bootstrap",
  reportfile = file.path(destination(repo), "manifest.html"),
  jsonrpt = TRUE)
```

Arguments

repo	A GRANRepository object
theme	CSS+JS theme. bootstrap, foundation, semanticui or jqueryui
reportfile	File path of the HTML report
jsonrpt	Whether to create a JSON version of the manifest report

Value

None

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

---

notifyManifest *Sends email notifications for a given manifest*

Description

Sends email notifications for a given manifest

Usage

```
notifyManifest(manifest, repo, ...)
```

Arguments

manifest	A dataframe obtained from getFailureInfo
repo	A gRAN repo object
...	Additonal arguments to GRANBase::sendMail()

Value

None

Note

This function is not intended for direct use by the end user.

See Also

sendMail for sending emails, isValidEmail for validating email, htmlTable for creating HTML tables

pkgHTML *Create HTML splash pages for packages*

Description

Create HTML splash pages for packages

Usage

```
pkgHTML(repo, splashname = "index.html", theme = "bootstrap")
```

Arguments

repo	A gRAN repo object
splashname	Filename for the HTML splash page
theme	CSS theme. bootstrap, foundation, semanticui or jqueryui

Value

None

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

readPkgsNEWS *Read and summarize the NEWS files for packages at risk (updatable)*

Description

readPkgsNEWS

Usage

```
readPkgsNEWS(df, oldlib = .libPaths(), tmplib = file.path(tempdir(),
  "libloc"), repos = unique(df$Repository), newlib = NULL)
```

Arguments

df	A data.frame or matrix of out-of-date packages currently installed, with columns Package, Installed (installed version), and Repository (contriburl of repo with newer version). Other columns are i gnored.
oldlib	The currently library to compare against latest avaialble versions
tmplib	A temporary library directory to install new versions of the packages into so that their NEWS files can be accessed.
repos	A character vector of the repositories to search for newer versions of packages installed in oldlib
newlib	An already populated 'new' library to compare against oldlib instead of retrieving new package versions from repos

Details

Attempts to generate a per-package summary of risky-to-ignore changes for updatable packages.

Value

A data.frame with 3 counts for each updatable package: bugfixes, u_visible_changes (user visible changes) and deprec (deprecation and defunct entries). All counts are NA if the package does not have parsable NEWS.

---

relatedPkgs *Wrapper for dependsOnPkgs*

Description

Wrapper for dependsOnPkgs

Usage

```
relatedPkgs(pkg_name, relation = "LinkingTo")
```

Arguments

pkg_name	The name of the package (string)
relation	What type of reverse dependency?

Value

String of reverse dependencies

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

reversals *Generate reverse dependency info as data.frame*

Description

Generate reverse dependency info as data.frame

Usage

```
reversals(pkg_name)
```

Arguments

pkg_name	The name of the package (string)

Value

data.frame containing package reverse dependency info

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

sendMail *Wrapper for sendmail, allows for multiple attachments to be sent*

Description

Wrapper for sendmail, allows for multiple attachments to be sent

Usage

```
sendMail(receiver, subject, body, repo, mailopts = email_options(repo),
  attachments = NULL)
```

Arguments

receiver	Receiver's email ID as a string. Vector if multiple email IDs
subject	Email subject as a string
body	Email message body as a string
repo	A gRAN repo object
mailopts	Email options as a list
attachments	Files with full paths as an array

Value

None

Note

This function is not intended for direct use by the end user.

See Also

`sendmail` for sending simple emails

`sortDelimitedString` *Alphabetically sort delimited strings*

Description

Alphabetically sort delimited strings

Usage

```
sortDelimitedString(string, delimiter = ",", ...)
```

Arguments

`string`	A string
`delimiter`	A delimiter that separates contents of string
`...`	Arguments that you want to pass to base::sort()

Value

String that is alphabetically sorted

Note

This function is not intended for direct use by the end user.

string2list *Converts delimited string to list*

Description

Converts delimited string to list

Usage

```
string2list(string, separator = ",")
```

Arguments

string	Input string
separator	The delimiter

Value

Processed list

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

testCoverage *Calculate and generate package code test coverage reports*

Description

Calculate and generate package code test coverage reports

Usage

```
testCoverage(repo, cores = 1)
```

Arguments

repo	A gRAN repo object
cores	How many CPU cores to use?

Value

repo A gRAN repo object with updated code coverage info

Author(s)

Dinakar Kulkarni <kulkard2@gene.com>

`updateArchive` *Move older package sources to the Archive directory*

Description

Move older versions of packages into the repo Archive

Usage

```
updateArchive(repo, repodest = destination(repo),
  archive = archivedir(repo), ext = "\\.tar\\..*$")
```

Arguments

`repo`	A GRAN repo object
`repodest`	The repo destination (something that looks like BASE_REPO_DIR/src/contrib)
`archive`	The Archive directory where older packages will be stored
`ext`	Regex describing the file extension of the built packages

Author(s)

Dinakar Kulkarni

Index