Package ‘GRANCore’

February 4, 2020

Type Package
Title Classes and Methods for ‘GRANBase’
Version 0.2.7
Author Gabriel Becker[aut,cre], Dinakar Kulkarni [aut,ctb]
Maintainer Gabriel Becker <gabembecker@gmail.com>
Copyright Genentech Inc
Description Provides the classes and methods for GRANRepository
objects that are used within the ‘GRAN’ build framework for R packages.
This is primarily used by the ‘GRANBase’ package and repositories
that are created by it.
License Artistic-2.0
Depends R (>= 3.1.0), switchr (>= 0.9.28), methods
URL https://github.com/gmbecker/GRANCore
BugReports https://github.com/gmbecker/GRANCore/issues
RoxygenNote 6.1.1
NeedsCompilation no
Repository CRAN
Date/Publication 2020-02-04 15:00:13 UTC

R topics documented:

  archivedir .......................... 2
  available.packages ........................ 3
  backup_archive ........................ 4
  checkout_dir ........................ 4
  check_result_dir ........................ 5
  contrib.url ........................ 5
  coverage_report_dir ........................ 6
  destination ........................ 6
  dest_base ........................ 7

1
archivedir

Return the full path to the archive directory for the final repository deployment.

Usage

```r
archivedir(repo)
```

Arguments

- `repo` a GRANRepository object

Value

The full path to the archive directory where the archived packages will be deployed to.
available.packages

available.packages A generic for available.packages and a method for GRANRepository objects

Description

available.packages A generic for available.packages and a method for GRANRepository objects

Usage

```r
available.packages(contriburl, method, fields = NULL,
                   type = getOption("pkgType"), filters = NULL, repos = NULL)
```

```r
# S4 method for signature 'ANY'
available.packages(contriburl, method, fields = NULL,
                   type = getOption("pkgType"), filters = NULL, repos = NULL)

# S4 method for signature 'GRANRepository'
available.packages(contriburl, method,
                   fields = NULL, type = getOption("pkgType"), filters = NULL, repos = NULL)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contriburl</td>
<td>The repository or contrib url</td>
</tr>
<tr>
<td>method</td>
<td>See base documentation</td>
</tr>
<tr>
<td>fields</td>
<td>See base documentation</td>
</tr>
<tr>
<td>type</td>
<td>The type of packages to query</td>
</tr>
<tr>
<td>filters</td>
<td>See base documentation</td>
</tr>
<tr>
<td>repos</td>
<td>Character string for the repository to query. GRANRepository objects should be passed to the contriburl argument.</td>
</tr>
</tbody>
</table>

Examples

```r
repo = GRANRepository(GithubManifest("gmbecker/fastdigest"), basedir = tempdir())
# none because the repository hasn't been built...
available.packages(repo)
```
**backup_archive**

Return path where packages are backed up by default when clearing the repo

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>backup_archive Return path where packages are backed up by default when clearing the repo</td>
</tr>
</tbody>
</table>

**Usage**

backup_archive(repo)

```r
## S4 method for signature 'GRANRepository'
backup_archive(repo)
```

**Arguments**

repo  
a GRANRepository object

**Value**

Directory where packages are backed up by default when clearing repo

---

**checkout_dir**

Return the directory that package sources will be checked-out into for use in the build process

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkout_dir Return the directory that package sources will be checked-out into for use in the build process</td>
</tr>
</tbody>
</table>

**Usage**

checkout_dir(repo)

```r
## S4 method for signature 'GRANRepository'
checkout_dir(repo)

## S4 method for signature 'NULL'
checkout_dir(repo)
```

**Arguments**

repo  
a GRANRepository object
**check_result_dir**

- **Value**
  
  For destination, the full path to the contrib directory the packages will be deployed to

- **Description**
  
  check_result_dir Return the path where check results for packages will be deployed for use in the build report.

- **Usage**
  
  ```r
  check_result_dir(repo)
  ```

  ```r
  ## S4 method for signature 'GRANRepository'
  check_result_dir(repo)
  ```

- **Arguments**
  
  - repo: a GRANRepository object

- **Value**
  
  The directory where check results should be deployed for use in the build report

**contrib.url**

- **Description**
  
  contrib.url A generic for contrib.url so that available.packages et al can interact with GRANRepository objects.

- **Usage**
  
  ```r
  contrib.url(repos, type = getOption("pkgType"))
  ```

  ```r
  ## S4 method for signature 'GRANRepository'
  contrib.url(repos,
              type = getOption("pkgType"))
  ```
Arguments

repos A repository to extract the contrib url from
type The type of package repository it is

Examples

```r
repo = GRANRepository(GithubManifest("gmbecker/fastdigest"), basedir = tempdir())
contrib.url(repo)
```

coverage_report_dir

```
coverage_report_dir(repo)  # S4 method for signature 'GRANRepository'
```

Arguments

repo a GRANRepository object

Value

The directory where test coverage results should be deployed for use in the build report

destination

```
destination(repo)  # S4 method for signature 'GRANRepository'
```

Description

destination Return the full path to the contrib directory for the final repository deployment.
**dest_base**

**Arguments**

repo a GRANRepository object

**Value**

For destination, the full path to the contrib directory the packages will be deployed to

---

**email_notify**

**Arguments**

repo a GRANRepository object

**Value**

For email_notify, should emails be sent for build failure notifications?

---

**Description**

**email_notify** Should emails be sent for build failure notifications?

**Usage**

email_notify(repo)

## S4 method for signature 'GRANRepository'

email_notify(repo)
Arguments
repo a GRANRepository object

Value
logical

email_options
email_options Email options for sending build failure notifications

Description
email_options Email options for sending build failure notifications

Usage
e-mail_options(repo)

## S4 method for signature 'GRANRepository'
e-mail_options(repo)

Arguments
repo a GRANRepository object

Value
A list containing the email options

erlogfile
Log file location of a GRAN (sub) repository

Description
erlogfile Retrieve the path to the errors-only log file for a GRAN repository

Usage
erlogfile(repo)

## S4 method for signature 'GRANRepository'
erlogfile(repo)

## S4 method for signature 'RepoBuildParam'
erlogfile(repo)
GRANRepository-class

Arguments
repo a GRANRepository object

Value
file location of the errors-only logfile

Description
A constructor for the GRANRepository class of S4 objects representing individual repositories

Usage
GRANRepository(manifest, results, param = RepoBuildParam(...), ...)

Arguments
manifest A PkgManifest object
results A data.frame containing previous build results
param A RepoBuildParam object controlling the location and behavior of the repository being built
... Passed through to the default value of param

Examples
man = GithubManifest("gmbecker/fastdigest")
repo = GRANRepository(man, basedir = tempdir())

install_packages,character,GRANRepository-method

Description
install_packages method for GRANRepository objects

Usage
## S4 method for signature 'character,GRANRepository'
install_packages(pkgs, repos,
    versions = NULL, verbose = FALSE, ...)
Arguments

pkgs  Character vector of package names to install
repos A GRANRepository
versions Ignored
verbose Whether verbose/debugging information should be printed
... Passed (eventually) to install.packages

Details

This method calls directly down to the character,character method of install_packages in switchr

install_result_dir

install_result_dir Return the path where instal results for packages will be deployed for use in the build report.

Description

install_result_dir Return the path where instal results for packages will be deployed for use in the build report.

Usage

install_result_dir(repo)

## S4 method for signature 'GRANRepository'
install_result_dir(repo)

Arguments

repo a GRANRepository object

Value

The directory where install results should be deployed for use in the build report
loadRepo  

*Backwards compatible load utility*

### Description
Load a repository serialized to an R code file
serialize a repository to a file so that it does not require GRANBase to load

### Usage

```r
loadRepo(filename)
```

```r
saveRepo(repo, filename)
```

### Arguments

- **filename**: The file to load
- **repo**: The GRANRepository object to save

### Examples

```r
repo = GRANRepository(GithubManifest("gmbecker/rpath"), basedir = tempdir())
fil = file.path(tempdir(), "repo.R")
saveRepo(repo, fil)
repo2 = loadRepo(fil)
```

logfile  

*Log file location of a GRAN (sub) repository*

### Description
Accessors

### Usage

```r
logfile(repo)
```

```r
## S4 method for signature 'GRANRepository'
logfile(repo)
```

```r
## S4 method for signature 'RepoBuildParam'
logfile(repo)
```

```r
## S4 method for signature 'GRANRepository'
dep_repos(x)
```
repo_name(x)

## S4 method for signature 'GRANRepository'
repo_name(x)

temp_repo(x)

## S4 method for signature 'GRANRepository'
temp_repo(x)

check_warn_ok(x)

## S4 method for signature 'GRANRepository'
check_warn_ok(x)

check_note_ok(x)

## S4 method for signature 'GRANRepository'
check_note_ok(x)

suspended_pkgs(x)

## S4 method for signature 'GRANRepository'
suspended_pkgs(x)

suspended_pkgs(x) <- value

## S4 replacement method for signature 'GRANRepository'
suspended_pkgs(x) <- value

## S4 method for signature 'GRANRepository'
sh_init_script(x)

## S4 replacement method for signature 'GRANRepository'
sh_init_script(x) <- value

extra_fun(x)

## S4 method for signature 'GRANRepository'
extra_fun(x)

check_test_on(x)

## S4 method for signature 'RepoBuildParam'
check_test_on(x)

## S4 method for signature 'GRANRepository'
check_test_on(x)
install_test_on(x)

## S4 method for signature 'RepoBuildParam'
install_test_on(x)

## S4 method for signature 'GRANRepository'
install_test_on(x)

## S4 method for signature 'GRANRepository'
logfun(x)

## S4 replacement method for signature 'GRANRepository'
logfun(x) <- value

use_cran_granbase(x)

use_cran_granbase(x) <- value

## S4 method for signature 'GRANRepository'
use_cran_granbase(x)

## S4 replacement method for signature 'GRANRepository'
use_cran_granbase(x) <- value

## S4 method for signature 'RepoBuildParam'
use_cran_granbase(x)

## S4 replacement method for signature 'RepoBuildParam'
use_cran_granbase(x) <- value

check_timeout(x)

check_timeout(x) <- value

## S4 method for signature 'GRANRepository'
check_timeout(x)

## S4 replacement method for signature 'GRANRepository'
check_timeout(x) <- value

## S4 method for signature 'RepoBuildParam'
check_timeout(x)

## S4 replacement method for signature 'RepoBuildParam'
check_timeout(x) <- value
build_timeout(x)

build_timeout(x) <- value

## S4 method for signature 'GRANRepository'
build_timeout(x)

## S4 replacement method for signature 'GRANRepository'
build_timeout(x) <- value

## S4 method for signature 'RepoBuildParam'
build_timeout(x)

## S4 replacement method for signature 'RepoBuildParam'
build_timeout(x) <- value

pkg_log_dir(x)

## S4 method for signature 'RepoBuildParam'
pkg_log_dir(x)

## S4 method for signature 'GRANRepository'
pkg_log_dir(x)

pkg_log_file(pkg, x)

## S4 method for signature 'ANY,RepoBuildParam'
pkg_log_file(pkg, x)

## S4 method for signature 'ANY,GRANRepository'
pkg_log_file(pkg, x)

platform(x)

platform(x) <- value

## S4 method for signature 'GRANRepository'
platform(x)

## S4 replacement method for signature 'GRANRepository'
platform(x) <- value

r_version(x)

r_version(x) <- value

## S4 method for signature 'GRANRepository'
r_version(x)
logfile

## S4 replacement method for signature 'GRANRepository'

```r
r_version(x) <- value

bioc_version(x)

bioc_version(x) <- value
```

## S4 method for signature 'GRANRepository'

```r
bioc_version(x)
```

## S4 replacement method for signature 'GRANRepository'

```r
bioc_version(x) <- value
```

### Arguments

- **repo**: a GRANRepository object
- **x**: A GRANRepository object
- **value**: The new parameter value
- **pkg**: The package name, accepted by pkg_log_file.

### Details

Set or retrieve the relevant values from a GRANRepository object.

These functions get or set individual repository build parameters on a GRANRepository object.

### Value

file location of the full logfile

### See Also

RepoBuildParam

### Examples

```r
repo = GRANRepository(GithubManifest("gmbecker/fastdigest"), basedir = tempdir())
# parameter object
param(repo)
## fundamental sub-objects
manifest(repo)
repo_results(repo)
## important directories
repobase(repo)
staging(repo)
temp_lib(repo)
otrack(repo)
destination(repo)
dest_base(repo)
windowsbindir(repo)
```
make_windows_bins

make_windows_bins

Description
make_windows_bins Return the logical that determines whether to build Windows binaries

Usage
make_windows_bins(repo)

## S4 method for signature 'GRANRepository'
make_windows_bins(repo)

Arguments
repo a GRANRepository object

Value
Logical indicating whether Windows binaries will be built
**metadatadir**

Return the full path to the pkg metadata directory for the final repository deployment.

### Description

metadatadir Return the full path to the pkg metadata directory for the final repository deployment.

### Usage

```r
metadatadir(repo)
```

### Arguments

- `repo` a GRANRepository object

### Value

The full path to the metadata directory

---

**notrack,GRANRepository-method**

Return the directory which stores retrieved versions of non-GRAN packages for use in virtual repositories

### Description

notrack Return the directory which stores retrieved versions of non-GRAN packages for use in virtual repositories

### Usage

```r
## S4 method for signature 'GRANRepository'
notrack(repo)
```

### Arguments

- `repo` a GRANRepository object

### Value

The path to the notrack directory
### param

**Extract parameter object**

**Description**

Extract parameter object

**Usage**

```r
param(x)

## S4 method for signature 'GRANRepository'
param(x)

param(x) <- value

## S4 replacement method for signature 'GRANRepository'
param(x) <- value
```

**Arguments**

- `x` An object with an associated parameter
- `value` A new parameter object

### pkg_doc_dir

**pkg_doc_dir** *Return the path where test coverage reports for packages will be deployed for use in the build report.*

**Description**

pkg_doc_dir Return the path where test coverage reports for packages will be deployed for use in the build report.

**Usage**

```r
pkg_doc_dir(repo)

## S4 method for signature 'GRANRepository'
pkg_doc_dir(repo)
```

**Arguments**

- `repo` a GRANRepository object

**Value**

The directory where test coverage results should be deployed for use in the build report
Description

Initialize the directory structure for the GRAN repo

Usage

prepDirStructure(basedir, repo_name, temp_repo, temp_checkout, tempLibLoc, destination, logfile, errlogfile)

Arguments

- `basedir`: The base directory. By default the temporary repository, temporary install library, and package staging area will be located in `<basedir>/<subrepoName>/`, while the temporary source checkout will be in the `basedir` itself.
- `repo_name`: The name of the repository, e.g. stable or devel
- `temp_repo`: Location to create the temporary repository
- `temp_checkout`: Location to create temporary checkouts/copies of package source code
- `tempLibLoc`: Location to create the temporary installed package library for use during the testing process
- `destination`: Base location (not including repository name) of the final repository to be built.
- `logfile`: Comprehensive GRAN log file
- `errlogfile`: Error log file

Note

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

Description

`repobase` Generic accessor function to retrieve the repo specific subdirectory within the base directory

Usage

repobase(repo)

```r
## S4 method for signature 'GRANRepository'
repobase(repo)
```
Arguments

repo a GRANRepository object

Value

The path to the repository specific directory

RepoBuildParam-class ResultsRow

Description

A row from the 'results' slot in the GRANRepository object

Parameters for building a GRAN repository. Most behavior during the GRAN building process is specified via this object/constructor.

Usage

ResultsRow(name = NA_character_, building = TRUE, status = "ok",
version = "0.0-0", lastAttempt = NA_character_,
lastAttemptVersion = NA_character_,
lastAttemptStatus = NA_character_, lastbuilt = NA_character_,
lastbuiltversion = NA_character_, lastbuiltstatus = NA_character_,
buildReason = NA_character_, maintainer = NA_character_,
suspended = FALSE)

RepoBuildParam(basedir, repo_name = "current",
temp_repo = file.path(basedir, repo_name, "tmprepo"),
temp_checkout = file.path(basedir, "tmpcheckout"),
errlog = file.path(basedir, repo_name, paste0("GRAN", repo_name, 
"-error.log")), logfile = file.path(basedir, repo_name, paste0("GRAN", repo_name, 
"-full.log")), check_note_ok = TRUE, check_warn_ok = TRUE,
tempLibLoc = file.path(basedir, repo_name, "LibLoc"),
extra_fun = function(...) NULL, destination = basedir, auth = "",
dest_url = makeFileURL(normalizePath2(destination)),
shell_init = character(), loginnerfun = writeGRANLog,
install_test = TRUE, check_test = TRUE, use_cran_granbase = TRUE,
archive_timing = 2, archive_retries = 2, build_timeout = 10 * 60,
check_timeout = 15 * 60, email_notifications = FALSE,
email_opts = list(smtp_server = "localhost", smtp_port = 25,
sender_email = paste0("gran", repo_name, "@localhost"), unsubscribers = 
NULL), repo_archive = file.path(destination, repo_name, "src", 
"contrib", "Archive"), repo_metadata_dir = file.path(destination, 
repo_name, "src", "contrib", "Meta"), make_windows_bins = TRUE,
platform = "", r_version = "", bioc_version = "")
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the package</td>
</tr>
<tr>
<td>building</td>
<td>Logical indicating whether the package is building</td>
</tr>
<tr>
<td>status</td>
<td>Status of the package build</td>
</tr>
<tr>
<td>version</td>
<td>Package version</td>
</tr>
<tr>
<td>lastAttempt</td>
<td>Last attempt time of package build</td>
</tr>
<tr>
<td>lastAttemptVersion</td>
<td>Package version when build was last attempted</td>
</tr>
<tr>
<td>lastAttemptStatus</td>
<td>Package build status of last attempt</td>
</tr>
<tr>
<td>lastbuilt</td>
<td>Last built time of package, if successful</td>
</tr>
<tr>
<td>lastbuiltversion</td>
<td>Last built version of package, if successful</td>
</tr>
<tr>
<td>lastbuiltstatus</td>
<td>Last built status of package, if successful</td>
</tr>
<tr>
<td>buildReason</td>
<td>The reason why the package build was attempted</td>
</tr>
<tr>
<td>maintainer</td>
<td>Package maintainer</td>
</tr>
<tr>
<td>suspended</td>
<td>Is the package suspended?</td>
</tr>
<tr>
<td>basedir</td>
<td>The base directory. By default the temporary repository, temporary install library, and package staging area will be located in &lt;basedir&gt;/&lt;subrepoName&gt;/, while the temporary source checkout will be in the basedir itself.</td>
</tr>
<tr>
<td>repo_name</td>
<td>The name of the repository, e.g. stable or devel</td>
</tr>
<tr>
<td>temp_repo</td>
<td>Location to create the temporary repository</td>
</tr>
<tr>
<td>temp_checkout</td>
<td>Location to create temporary checkouts/copies of package source code</td>
</tr>
<tr>
<td>errlog</td>
<td>The file to append error output to during the building and testing processes</td>
</tr>
<tr>
<td>logfile</td>
<td>The file to append summary log information to during building and testing</td>
</tr>
<tr>
<td>check_note_ok</td>
<td>logical. Whether packages that raise notes during R CMD check should be considered to have passed</td>
</tr>
<tr>
<td>check_warn_ok</td>
<td>logical. Whether packages that raise warnings during R CMD check should be considered to have passed</td>
</tr>
<tr>
<td>tempLibLoc</td>
<td>Location to create the temporary installed package library for use during the testing process</td>
</tr>
<tr>
<td>extra_fun</td>
<td>currently ignored</td>
</tr>
<tr>
<td>destination</td>
<td>Base location (not including repository name) of the final repository to be built. NOTE: GRANBase:::buildReportURL() expects this argument to have a trailing slash.</td>
</tr>
<tr>
<td>auth</td>
<td>character. Authentication information required to add packages to the manifest.</td>
</tr>
<tr>
<td>dest_url</td>
<td>The base URL the destination directory corresponds to. The subrepository name will be appended to this to generate the URL used when installing from the repository.</td>
</tr>
</tbody>
</table>
shell_init  
An optional shell script to source before invoking system commands, e.g. a bashrc file. Ignored if "" or not specified.

loginnerfun  
The function to use to write log messages during the repository build process. It will be passed pkg, ..., errfile, logfile, and pkglog based on the other arguments to this function. Defaults to writeGRANLog specified as the full and error log locations, respectively.

install_test  
logical. Should the install test be performed? Required to build packages with vignettes, and for the check test

check_test  
logical. Should R CMD check be run on the packages as a cohort. Requires install test.

use_cran_granbase  
logical. Currently ignored.

archive_timing  
numeric. Number of seconds to wait between attempts to pull a package from the CRAN archive

archive_retries  
numeric. Number of times to retry pulling a package from the CRAN archive.

build_timeout  
numeric. Number of seconds before timeout during the build step for a single package. Defaults to 10 minutes.

check_timeout  
numeric. Number of seconds before timeout during the check step for a single package. Defaults to 15 minutes.

email_notifications  
logical. Should email notifications be sent regarding packages that fail to build on GRAN? Defaults to FALSE

email_opts  
List. Email options for sending emails regarding packages that fail to build on GRAN. The list contains 4 elements: smtp_server: the SMTP server - defaults to "localhost", smtp_port: SMTP port number - defaults to 25, sender_email: Whom should the emails be sent as? Defaults to "gran<repo_name>@localhost", unsubscribers: Vector of Perl-style regexes for unsubscribers - defaults to NULL.

repo_archive  
Archive directory where older package sources will be saved

repo_metadata_dir  
Directory containing metadata files

make_windows_bins  
Whether to make Windows binary packages

platform  
Name of platform this GRANRepository is being build on

r_version  
R version for this GRANRepository

bioc_version  
Bioconductor version for this GRANRepository

Value  

data.frame

Note  

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

*Transform a GRANRepository object into a list*

**Examples**

```r
case = ResultsRow(name = "mypkg")
b = RepoBuildParam(basedir = tempdir(), repo_name = "myrepo")
```

**Description**

Utility to transform a GRANRepository object into a list so that repos saved using GRANBase can be loaded by GRAN without requiring GRANBase

**Usage**

```r
RepoToList(repo)
RepoFromList(rlist)
```

**Arguments**

- **repo**: repository
- **rlist**: A list with entries that are slot name-value pairs for a GRANRepository object

**Value**

- a list suitable for use with RepoFromList
- a GRANRepository object

**Examples**

```r
case = GRANRepository(GithubManifest("gmbecker/switchr"), basedir = tempdir())
lst = RepoToList(case)
c = RepoFromList(lst)
```

**repo_results**: *Repository build results*

**Description**

Repository build results
Usage

repo_results(x)

## S4 method for signature 'GRANRepository'
repo_results(x)

repo_results(x) <- value

## S4 replacement method for signature 'GRANRepository'
repo_results(x) <- value

Arguments

x A GRANRepository object
value The new results data.frame

Value

A data.frame of build results

desc repo_url

repo_url(repo)

## S4 method for signature 'GRANRepository'
repo_url(repo)

## S4 method for signature 'NULL'
repo_url(repo)

Arguments

repo a GRANRepository object

Value

For destination, the full path to the contrib directory the packages will be deployed to
staging

Return the staging directory or the staging_logs to be used when building the repository. If the directory does not exist it will be created.

Description

staging Return the staging directory or the staging_logs to be used when building the repository. If the directory does not exist it will be created.

Usage

staging(repo)

## S4 method for signature 'GRANRepository'
staging(repo)
staging_logs(repo)

## S4 method for signature 'GRANRepository'
staging_logs(repo)

Arguments

repo a GRANRepository object

Value

The path to the repository specific directory
The path to the repository specific directory

temp_lib temporary library

description

temporary library

Usage
temp_lib(repo)

## S4 method for signature 'GRANRepository'
temp_lib(repo)
Arguments

repo A GRANRepository object

updateGRANRepoObject  

Description

Update the GRAN repo object with newer information. This is called internally by loadRepo to ensure that the resulting repo doesn’t have stale logging closures and the like.

Usage

updateGRANRepoObject(object, ...)

Arguments

object The GRAN repo object
...

Other parameters from RepoBuildParam

Note

This function is not intended for use by the end user. This is called internally by loadRepo.

windowsbindir  

windowsbindir Return the full path to the location of the windows binary builds

Description

windowsbindir Return the full path to the location of the windows binary builds

Usage

windowsbindir(repo)

## S4 method for signature 'GRANRepository'
windowsbindir(repo)

Arguments

repo a GRANRepository object

Value

Full path to the location of the windows binary builds
**Description**

Utility function which writes gran logs

**Usage**

```r
writeGRANLog(pkg, msg, type = "full", logfile, errfile, pkglog = NULL)
```

**Arguments**

- `pkg` The name of the package the log is about
- `msg` The log message, collapsed if length>1
- `type` "full", "error", "warn", or "both" indicating which log(s) the message should be written to
- `logfile` The location of the full log file to write/append to
- `errfile` The location of the error log file to write/append to
- `pkglog` character. The package-specific log file to write to if applicable.

**Note**

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

archivedir, 2
archivedir, GRANRepository-method (archivedir), 2
available.packages, 3
available.packages, ANY (available.packages), 3
available.packages, ANY-method (available.packages), 3
available.packages, GRANRepository (available.packages), 3
available.packages, GRANRepository-method (available.packages), 3
backup_archive, 4
backup_archive, (backup_archive), 4
backup_archive, GRANRepository-method (backup_archive), 4
bioc_version (logfile), 11
bioc_version, GRANRepository (logfile), 11
bioc_version, GRANRepository-method (logfile), 11
bioc_version<-, (logfile), 11
bioc_version<-, GRANRepository (logfile), 11
bioc_version<-, GRANRepository-method (logfile), 11
build_timeout (logfile), 11
build_timeout, GRANRepository (logfile), 11
build_timeout, GRANRepository-method (logfile), 11
build_timeout<-, GRANRepository-method (logfile), 11
build_timeout<-, RepoBuildParam (logfile), 11
build_timeout<-, RepoBuildParam-method (logfile), 11
check_note_ok (logfile), 11
check_note_ok, GRANRepository (logfile), 11
check_note_ok, GRANRepository-method (logfile), 11
check_result_dir, 5
check_result_dir, GRANRepository-method (check_result_dir), 5
check_test_on (logfile), 11
check_test_on, GRANRepository (logfile), 11
check_test_on, GRANRepository-method (logfile), 11
check_test_on, RepoBuildParam (logfile), 11
check_test_on, RepoBuildParam-method (logfile), 11
check_timeout (logfile), 11
check_timeout, GRANRepository (logfile), 11
check_timeout, GRANRepository-method (logfile), 11
check_timeout<-, (logfile), 11
check_timeout<-, GRANRepository (logfile), 11
check_timeout<-, RepoBuildParam-method (logfile), 11
check_timeout<-, RepoBuildParam (logfile), 11
check_timeout<-,RepoBuildParam-method
  (logfile), 11
check_warn_ok (logfile), 11
check_warn_ok,GRANRepository (logfile), 11
check_warn_ok,GRANRepository-method
  (logfile), 11
cHECKout_dir, 4
cHECKout_dir,GRANRepository-method
  (checkout_dir), 4
cHECKout_dir,GRANRepository-method
  (checkout_dir), 4
cHECKout_dir,NULL (checkout_dir), 4
cHECKout_dir,NULL-method
  (checkout_dir), 4
cOntrib.url, 5
cOntrib.url,GRANRepository
  (Contrib.url), 5
cOntrib.url,GRANRepository-method
  (Contrib.url), 5
cOVERage_report_dir, 6
cOVERage_report_dir,GRANRepository-method
  (coverage_report_dir), 6
dEP_repos,GRANRepository-method
  (logfile), 11
dEP_repos,RepoBuildParam (logfile), 11
dEST_base, 7
dEST_base,GRANRepository-method
  (dest_base), 7
dESTination, 6
dESTination,GRANRepository-method
  (destination), 6
eMAIL_notify, 7
eMAIL_notify,GRANRepository-method
  (email_notify), 7
eMAIL_options, 8
eMAIL_options,GRANRepository-method
  (email_options), 8
errologfile, 8
errologfile,GRANRepository-method
  (errologfile), 8
errologfile,RepoBuildParam-method
  (errologfile), 8
eXTRA Functor (logfile), 11
eXTRA Functor,GRANRepository-method
  (logfile), 11
GRANRepository (GRANRepository-class), 9
GRANRepository-class, 9
GRANRepository-method (backup_archive), 4
install_packages,character,GRANRepository-method, 9
install_result_dir, 10
install_result_dir,GRANRepository-method
  (install_result_dir), 10
install_test_on (logfile), 11
install_test_on,GRANRepository
  (logfile), 11
install_test_on,GRANRepository-method
  (logfile), 11
install_test_on,RepoBuildParam
  (logfile), 11
install_test_on,RepoBuildParam-method
  (logfile), 11
loadRepo, 11
logfile, 11
logfile,GRANRepository-method
  (logfile), 11
logfile,RepoBuildParam-method
  (logfile), 11
logfile-method,GRANRepository
  (logfile), 11
logfile,GRANRepository
  (logfile), 11
logfile,GRANRepository-method
  (logfile), 11
logfile,GRANRepository-method
  (logfile), 11
make_windows_bins, 16
make_windows_bins,GRANRepository-method
  (make_windows_bins), 16
metadatadir, 17
metadatadir,GRANRepository-method
  (metadatadir), 17
notrack,GRANRepository-method, 17
param, 18
param,GRANRepository (param), 18
param,GRANRepository-method (param), 18
param<-(param), 18
param<-,GRANRepository (param), 18
param<-,GRANRepository-method (param), 18
INDEX

temp_repo,GRANRepository (logfile), 11
  temp_repo,GRANRepository-method
    (logfile), 11

updateGRANRepoObject, 26
use_cran_granbase (logfile), 11
use_cran_granbase,GRANRepository
  (logfile), 11
use_cran_granbase,GRANRepository-method
  (logfile), 11
use_cran_granbase,RepoBuildParam
  (logfile), 11
use_cran_granbase,RepoBuildParam-method
  (logfile), 11
use_cran_granbase<-(logfile), 11
use_cran_granbase<-,GRANRepository
  (logfile), 11
use_cran_granbase<-,GRANRepository-method
  (logfile), 11
use_cran_granbase<-,RepoBuildParam
  (logfile), 11
use_cran_granbase<-,RepoBuildParam-method
  (logfile), 11

windowsbindir, 26
windowsbindir,GRANRepository-method
  (windowsbindir), 26
writeGRANLog, 27