Package ‘switchr’

April 20, 2020

Type Package

Title Installing, Managing, and Switching Between Distinct Sets of Installed Packages

Version 0.14.3

Author Gabriel Becker[aut, cre]

Maintainer Gabriel Becker <gabembecker@gmail.com>

Copyright Genentech Inc

Description Provides an abstraction for managing, installing, and switching between sets of installed R packages. This allows users to maintain multiple package libraries simultaneously, e.g. to maintain strict, package-version-specific reproducibility of many analyses, or work within a development/production release paradigm. Introduces a generalized package installation process which supports multiple repository and non-repository sources and tracks package provenance.

Imports tools, RJSONIO, RCurl

Depends methods

SystemRequirements git, svn

License Artistic-2.0

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

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

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-04-20 08:30:02 UTC

R topics documented:

`.libPaths2` *.libpaths2*

Description

A version of .libPaths which allows for excluding the site library

Usage

```
.libPaths2(fulllp, exclude.site = TRUE)
```

Arguments

`fulllp`	The libpath to use, as in .libPaths
`exclude.site`	logical. Should the site library be suppressed. Defaults to TRUE

Details

Behaves exactly as the .libPaths function does, with the exception of optionally excluding the site library

---

addPkg *addPkg*

Description

Add a package to an object associated with a manifest

Usage

```
addPkg(x, ..., rows = makeManifest(...), versions = data.frame(name =
  manifest_df(rows)$name, version = NA_character_, stringsAsFactors =
  FALSE), replace = FALSE)

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

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

Arguments

x	A manifest or manifest-associate objec tto add the pkg 2
...	The information regarding the package to place in the manifest
rows	An already-created data.frame to add to the manifest
versions	A data.frame of package names and versions, if adding to a SessionManifest, ignored otherwise
replace	logical. If true, the specified package info will replace any already in the manifest in the case of duplicates. Otherwise, an error is thrown.

`archive_retries` *archive_retries*

Description

Get or set the number of times to retry downloading a file from the CRAN archive

This is intended to stop intermittent install failures due to failing to retrieve files that *are* in the archive but are not downloading properly when a larger number of packages is being retrieved.

Usage

```
archive_retries(x)

## S4 method for signature 'SwitchrParam'
archive_retries(x)

archive_retries(x) <- value

## S4 replacement method for signature 'SwitchrParam'
archive_retries(x) <- value
```

Arguments

`x`	A SwitchrParam object
`value`	The new number of seconds to wait

Value

When getting, the number of seconds to wait, when setting, a new, updated SwitchrParam object.

`archive_timing` *archive_timing*

Description

Get or set the number of seconds to wait after trying to retrieve a file from the CRAN Archive.

This is intended to stop intermittent install failures due to failing to retrieve files that *are* in the archive but are not downloading properly when a larger number of packages is being retrieved.

Usage

```
archive_timing(x)

## S4 method for signature 'SwitchrParam'
archive_timing(x)

archive_timing(x) <- value

## S4 replacement method for signature 'SwitchrParam'
archive_timing(x) <- value
```

Arguments

`x`	A SwitchrParam object
`value`	The new number of seconds to wait

Value

When getting, the number of seconds to wait, when setting, a new, updated SwitchrParam object.

`BiocDevel` *BiocDevel*

Description

An object representing the current Bioc devel version. Can be passed to switchTo.

Usage

```
BiocDevel
```

Format

An object of class `RepoSubset` of length 1.

`BiocRelease` *BiocRelease*

Description

An object representing the current Bioc release. Can be passed to switchTo.

Usage

```
BiocRelease
```

Format

An object of class `RepoSubset` of length 1.

`biocReposForVers` *biocReposForVers*

Description

Generate the URLs of the repositories associated with a specific Bioconductor release

Usage

```
biocReposForVers(version)
```

Arguments

`version`	The Bioconductor release to generate URLs for.

Note

This function will only work if some version of Bioconductor (>2.9) was installed when switchr was installed. It will return NULL otherwise.

BiocSVNManifest *DEPCRECATED - Create a manifest of Bioc SVN locations*

Description

DEPCRECATED - Create a manifest of Bioc SVN locations

Usage

```
BiocSVNManifest(bioc_vers = "devel", software_only = TRUE)
```

Arguments

`bioc_vers`	A version number for a bioc release, or `"devel"` to for the current devel trunk
`software_only`	logical. Should only software packages be included in the manifest? Defaults to TRUE

Details

In combination with the `lazyRepo` function, this manifest can be used to work from a local, working checkout of a set of inter-dependent Bioconductor packages.

Value

A PkgManifest which contains SVN locations for all packages found in the specified bioc repositories, as well as those listed in `not_in_repo`

See Also

`lazyRepo`

BiocVers *BiocVers*

Description

A constructor for creating a RepoSubset object for a specified release of Bioconductor, which includes only the BiocInstaller package.

Usage

```
BiocVers(version = getBiocReleaseVr(), name = paste("BioC", version,
  sep = "_"), repos = biocReposForVers(version))
```

Arguments

`version` The version of Bioconductor

`name` The default name for switchr libraries created with this object

`repos` The urls of the Bioconductor repositories. these will be modified automatically to match the specified version

branch *branch*

Description

Get or set the branch associated with a Package Source

Usage

```
branch(x)

## S4 method for signature 'PkgSource'
branch(x)

branch(x) <- value

## S4 replacement method for signature 'PkgSource'
branch(x) <- value
```

Arguments

`x` A source

`value` The new branch

c,SessionManifest-method *cmethods*

Description

Combine 2 or more manifests of the same type (PkgManifest or SessionManifest)

Usage

```
## S4 method for signature 'SessionManifest'
c(x, ..., recursive = FALSE)

## S4 method for signature 'PkgManifest'
c(x, ..., recursive = FALSE)
```

Arguments

`x`	An object (indicates the type of all objects to be combined)
`...`	more objects
`recursive`	Unused

`checkIsPkgDir` *Check if a directory contains package sources*

Description

Check if a directory contains package sources

Usage

```
checkIsPkgDir(dir)
```

Arguments

`dir`	The directory

`cranPkgVersManifest` *cranPkgVersManifest*

Description

Create a Pkg manifest which points to tarballs representing a particular version of a CRAN package and versions of its (recursive) dependencies that were contemporary on the first or last day the specified package version resided on CRAN

Usage

```
cranPkgVersManifest(pkg, vers, earliest = TRUE,
  cur_avail = available.packages(), verbose = FALSE,
  suggests = c("direct", "none"), delay = 1, erronfail = TRUE)
```

Arguments

`pkg`	The package on which to base the generated manifest
`vers`	The version of pkg to construct the cohort around. Note this must match the the version string exactly, i.e. 1.3.1 and 1.3-1 are *not* equivalent.
`earliest`	Should the package dependencies be contemporary with the first (TRUE) or last (FALSE) day the specified package version was (the latest version) on CRAN?
`cur_avail`	The output from available.packages(). Used to identify whether the necessary version is in the CRAN archive or normal repository

verbose	Should debugging information about the recursive traversal of package dependencies be printed (defaults to FALSE).
suggests	Which Suggests'ed packages should be included. Currently supported possibilites are direct, indicating Suggestions of pkg should be included, and none, indicating that no Suggests'ed packages should be counted.
delay	Number of seconds to delay between successive REST calls to the crandb database. Defaults to 1 second
erronfail	how should connection errors be handled. TRUE (the default) throws an error, NA throws a warning, FALSE emits a message.

Value

A SessionManifest object

Note

Some packages retain the same version on CRAN for long periods of time. The cohort in the manifest represents a gross proxy for the cohort used in conjunction within an analysis which used a the vers version of the specified package. In general it will *not* perfectly recreate the set of package versions originally used.

Author(s)

Gabriel Becker

References

"Gabor Csardi" (2014). crandb: Query the unofficial CRAN metadata database. R package version 1.0.0. https://github.com/metacran/crandb

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

Examples

```
## Not run:
man = cranPkgVersManifest("devtools", "1.6")

## End(Not run)
```

currentCompEnv *currentCompEnv*

Description

Display the computing environment currently in use. If switchTo has not been called, a new SwitchrCtx object describing the current environment is created.

Usage

```
currentCompEnv()
```

---

defaultRepos *defaultRepos*

Description

Get default repositories for use as dependency repos and within install_packages

Usage

```
defaultRepos()
```

Value

A character vector of package repository urls

---

dep_repos *dep_repos*

Description

Get or set repositories to be used to fullfill dependencies beyond packages within the manifest

Usage

```
dep_repos(x)

## S4 method for signature 'PkgManifest'
dep_repos(x)

## S4 method for signature 'SessionManifest'
dep_repos(x)

dep_repos(x) <- value

## S4 replacement method for signature 'PkgManifest'
dep_repos(x) <- value

## S4 replacement method for signature 'SessionManifest'
dep_repos(x) <- value
```

Arguments

`x`	A package or session manifest
`value`	A character vector with the new dependency repos

Value

Character vector with existing repository urls

`dl_method` *dl_method*

Description

Get or set the download method for retreiving files.

Usage

```
dl_method(x)

## S4 method for signature 'SwitchrParam'
dl_method(x)

dl_method(x) <- value

## S4 replacement method for signature 'SwitchrParam'
dl_method(x) <- value
```

Arguments

`x`	A SwitchrParam object
`value`	The new number of seconds to wait

`errorOrNonZero` *Identify error states from R or external programs*

Description

Identify error states from R or external programs

Usage

```
errorOrNonZero(out)
```

Arguments

`out`	An R object representing output

Value

TRUE if out is an error object, or has an attribute called "status" which is > 0

fileFromFileURL *Get path from file URL*

Description

Get path from file URL

Usage

```
fileFromFileURL(fileurl)
```

Arguments

fileurl	A file url (beginning in file://)

Value

The system directory path that `fileurl` points to

findNewestPkgInds *Find newest packages in a package info data.frame*

Description

Find newest packages in a package info data.frame

Usage

```
findNewestPkgInds(df, pkgcol = "package", verscol = "version")

findNewestPkgRows(df, pkgcol = "package", verscol = "version",
  newcol = "new", verbose = FALSE, logfun = message)
```

Arguments

df	data.frame. Table of package information
pkgcol	string. Name of column containing package name
verscol	string. Name of column containing package version in version-string form.
newcol	character. Experimental. column name for the column indicating that the version is new.
verbose	logical. Should debugging information be written using `logfun` during this process.
logfun	function. Logging function (closure) which should be called to write verbose logging messages during the process.

Value

a data.frame with the same columns as df which contains only the most recent row for each unique package name, as determined by the contents of df[[verscol]]

for findNewestPkgInds, the indices of the rows representing the newest version of each package within df. For findNewestPkgRows, the rows themselves from df representing the newest version of each package.

---

findPkgDir *Find a package directory within an SCM checkout*

Description

Find a package directory within an SCM checkout

Usage

```
findPkgDir(rootdir, branch, subdir, param)
```

Arguments

rootdir	The directory of the checkout
branch	The branch to navigate to
subdir	The subdirectory to navigate to
param	a SwitchrParam object

Value

A path to the Package sources

---

findPkgVersionInRepo *findPkgVersionInRepo*

Description

findPkgVersionInRepo

Usage

```
findPkgVersionInRepo(repo, name, version, param, dir)

## S4 method for signature 'character'
findPkgVersionInRepo(repo, name, version, param, dir)

## S4 method for signature '`NULL`'
findPkgVersionInRepo(repo, name, version, param, dir)
```

Arguments

`repo`	The repository
`name`	The name of the package
`version`	The version of the package to find
`param`	A SwitchrParam object
`dir`	The directory to download the located package tarball into

Value

A path to the downloaded tarball, or NULL

`flushSession` *flushSession*

Description

Unload currently loaded packages from the current R session

Usage

```
flushSession(dontunload = switchrDontUnload())
```

Arguments

`dontunload`	Non-base packages to ignore (not detatch/unload)

Details

Attached packages are detached (and unloaded) first. After this is done, loaded packages, such as those imported by (previously) attached packages, are unloaded.

Finally, after all packages have been unloaded, native libraries loaded by those packages are unloaded (on systems where this is supported).

Value

NULL, called for its side-effect of unloading packages

Note

Failing to include switchr, any of its dependencies, or any base packages (available as a vector in the `switchDeps` object) in `dontunload` will result in undefined, likely erroneous behavior.

`full_libpaths` *full_libpaths*

Description

Accessor for the full library path associate with a SwitchrCtx, including the R library and (if not excluded) the site library

Usage

```
full_libpaths(seed)

## S4 method for signature 'SwitchrCtx'
full_libpaths(seed)
```

Arguments

`seed`	a SwitchrCtx

`getPkgDir` *Construct pockage directory path*

Description

Construct pockage directory path

Usage

```
getPkgDir(basepath, name, subdir, scm_type, branch)
```

Arguments

`basepath`	The parent directory for the package directory
`name`	The name of the package
`subdir`	The subdirectory within a package source that the actual package root directory will reside in.
`scm_type`	Tye type of scm the package sources will be checked out from
`branch`	The branch from which the package will be retrieved.

Value

A path

Note

Unlike `findPkgDir` this does not look for existing package source directories. It only constructs the path.

GithubManifest *GithubManifest*

Description

Create a package manifest containing only github packages

Usage

```
GithubManifest(..., pkgrepos)
```

Arguments

`...`	Combined to populate `pkgrepos`
`pkgrepos`	Github repositories in the form "<user>/<reponame>"

Details

Any names of the pkgrepos vector are assumed to be pkg names for the manifest. For unnamed elements, the pkg name is assumed to be the repository name.

Note

This is a convenience wrapper for `makeManifest`. It uses the `username/repo[/subdir][@ref]` shorthand for specifying package locations in github repositories introduced by Wickham's devtools. Unlike devtools, username is not optional, and only branch names are currently supported in the `@ref`

Examples

```
ghman = GithubManifest("gmbecker/switchr", "hadley/devtools")
ghman
```

gotoVersCommit *gotoVersCommit*

Description

This is a low-level function not intended for direct use by the end user.

Usage

```
gotoVersCommit(dir, src, version, param = SwitchrParam())

## S4 method for signature 'character,SVNSource'
gotoVersCommit(dir, src, version,
  param = SwitchrParam())

## S4 method for signature 'character,CRANSource'
gotoVersCommit(dir, src, version,
  param = SwitchrParam())

## S4 method for signature 'character,BiocSource'
gotoVersCommit(dir, src, version,
  param = SwitchrParam())

## S4 method for signature 'character,GitSource'
gotoVersCommit(dir, src, version,
  param = SwitchrParam())
```

Arguments

`dir`	Directory
`src`	A PkgSource (or subclass) object
`version`	The exact version to locate
`param`	A SwitchrParam

`graceful_inet` *Internal internet harness*

Description

This function should never be called by code outside of tests/vignettes in this package or packages that depend on it.

Usage

```
graceful_inet(val)

warning2(...)
```

Arguments

`val`	logical. NA means no additional handling, TRUE, means careful handling but actually attempt the call, FALSE means force immedate failure without evaling expressions wrapped in inet_handlers() calls
`...`	passed to message or base::warning

head *Head and tail operations on manifests*

Description

Head and tail operations on manifests

Usage

```
head(x, ...)

## S4 method for signature 'SessionManifest'
head(x, n = 5, ...)

## S4 method for signature 'PkgManifest'
head(x, n = 5, ...)

tail(x, ...)

## S4 method for signature 'SessionManifest'
tail(x, n = 5, ...)

## S4 method for signature 'PkgManifest'
tail(x, n = 5, ...)
```

Arguments

`x`	A manifest object
`...`	unused
`n`	The number of packages to keep

Details

In the case of a `PkgManifest`, the first or last `n` packages are retained in the manifest, while all others are removed.

In the case of a `SessionManifest`, `n` specified versions are retained, while the underlying `PkgManifest` is unchanged.

Value

An object of the same type as `x` containing `n` packages

`install_packages` *install_packages*

Description

Install packages from a set of traditional repositories, or a Just-in-time repository constructed using a PkgManifest or SessionManifest

Usage

```
install_packages(pkgs, repos, versions = NULL, verbose = FALSE, ...)

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

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

## S4 method for signature 'SessionManifest,ANY'
install_packages(pkgs, repos,
  versions = NULL, verbose = FALSE, ...)

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

## S4 method for signature 'character,PkgManifest'
install_packages(pkgs, repos,
  versions = NULL, verbose = FALSE, ...)
```

Arguments

`pkgs`	The names of the packages to install
`repos`	The (generalized) repositor(ies) to install the packages from. Can be a character vector of traditional package repositories (as with install.packages) or a PkgManifest or SessionManifest (or a url thereof)
`versions`	An optional named character vector or data.frame specifying exact versions of the packages to install
`verbose`	Should extra information be printed during the console during installation
`...`	extra parameters passed directly to install.packages

Details

In addition to installing the specified packages, this function annotates the installed DESCRIPTION files with provenance information about where the packages were installed from. This retains the information necessary to generate a manifest of installed packages for publication or reinstallation.

When repos is a vector of traditional repositories, this function - with the exception of the provenance mentioned above - behaves identically to install.packages. Otherwise, a Just-in-Time package repository is constructed using the information in the manifest(s) passed to repos, which is then used in conjuction with link{install.packages} to do the actual installation.

Author(s)

Gabriel Becker

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

Examples

```
## Not run:
## equivalent to install.packages, except it stores
## package provenance and knows about bioconductor repos
install_packages("nlme")

## install from a manifest
man = GithubManifest("gmbecker/fastdigest")
install_packages("fastdigest", man)

## install a full seeding manifest
man2 = makeSeedMan("myotherlib")
install_packages(man2)

## End(Not run)
```

lazyRepo *lazyRepo*

Description

Create a lazy repository for installing directly from a package manifest. Most users will want to call Install directly, which will call this as needed behind the scenes.

Usage

```
lazyRepo(pkgs, pkg_manifest, versions = rep(NA, times = length(pkgs)),
  dir = tempdir(), rep_path = file.path(dir, ”repo”),
  get_suggests = FALSE, verbose = FALSE,
  scm_auths = list(bioconductor = c(”readonly”, ”readonly”)),
  param = SwitchrParam(), force_refresh = FALSE)

## S4 method for signature 'SessionManifest,ANY'
lazyRepo(pkgs, pkg_manifest,
  versions = rep(NA, times = length(pkgs)), dir = tempdir(),
  rep_path = file.path(dir, ”repo”), get_suggests = FALSE,
  verbose = FALSE, scm_auths = list(bioconductor = c(”readonly”,
  ”readonly”)), param = SwitchrParam(), force_refresh = FALSE)

## S4 method for signature 'PkgManifest,ANY'
lazyRepo(pkgs, pkg_manifest,
  versions = rep(NA, times = length(pkgs)), dir = tempdir(),
  rep_path = file.path(dir, ”repo”), get_suggests = FALSE,
  verbose = FALSE, scm_auths = list(bioconductor = c(”readonly”,
  ”readonly”)), param = SwitchrParam(), force_refresh = FALSE)

## S4 method for signature 'character,SessionManifest'
lazyRepo(pkgs, pkg_manifest,
  versions = rep(NA, times = length(pkgs)), dir = tempdir(),
  rep_path = file.path(dir, ”repo”), get_suggests = FALSE,
  verbose = FALSE, scm_auths = list(bioconductor = c(”readonly”,
  ”readonly”)), param = SwitchrParam(), force_refresh = FALSE)

## S4 method for signature 'character,PkgManifest'
lazyRepo(pkgs, pkg_manifest,
  versions = rep(NA, times = length(pkgs)), dir = tempdir(),
  rep_path = file.path(dir, ”repo”), get_suggests = FALSE,
  verbose = FALSE, scm_auths = list(bioconductor = c(”readonly”,
  ”readonly”)), param = SwitchrParam(), force_refresh = FALSE)
```

Arguments

`pkgs` The packages to install

`pkg_manifest` The manifest to use

`versions` Specific versions of the packages to install. Should be a vector of the same length as pkgs (and in the same order). Defaults to NA (any version) for all packages.

`dir` The directory packages should be downloaded/checkedout/built into

`rep_path` The path of the final repository

`get_suggests` Whether suggested packages should be included in the lazy repository. Defaults to FALSE

`verbose` Should extra information be printed to the user during the construction process `scm_auths` Named list of username/password credentials for checking out package sources from one or more sources listed in `manifest` Defaults to readonly access to Bioconductor SVN

`param` A SwitchrParam object

`force_refresh` If a package already appears in the lazy repo area, it be updated (e.g. from SCM) and built again? Defaults to FALSE

Details

When checking building from SVN or git checkouts, this function will first look for existing checkouts for the relevant packages in `dir`. If found, these will be updated (in the case of conflicts, the behavior is undefined and will likely fail if they are not resolvable). This allows the user to have an existing, checkout directory where he or she works on development versions of multiple, interrelated packages, as local changes WILL be reflected in the packages built into the lazy repository.

Value

A path to the populated lazy repository, suitable for 'coercing' to a url and installing from.

Author(s)

Gabriel Becker

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

`libManifest` *libManifest*

Description

Create a Session- or PkgManifest for the contents of a switchr library.

Usage

```
libManifest(lib = currentCompEnv(), record_versions = TRUE,
  known_manifest = makeManifest(dep_repos = repos),
  repos = defaultRepos(), ...)

## S4 method for signature 'missing'
libManifest(lib = currentCompEnv(),
  record_versions = TRUE, known_manifest = makeManifest(dep_repos =
  repos), repos = defaultRepos(), ...)

## S4 method for signature 'character'
```

```
libManifest(lib = currentCompEnv(),
  record_versions = TRUE, known_manifest = makeManifest(dep_repos =
  repos), repos = defaultRepos(), ...)

## S4 method for signature 'SwitchrCtx'
libManifest(lib = currentCompEnv(),
  record_versions = TRUE, known_manifest = makeManifest(dep_repos =
  repos), repos = defaultRepos(), ...)
```

Arguments

`lib`	A SwitchrCtx object, or the name of a switchr library. Defaults to the currently active switchr library.
`record_versions`	Should the exact versions of installed packages be recorded in the manifest (TRUE)
`known_manifest`	An existing manifest, used when imputing location information for packages not installed via `install_packages`
`repos`	A vector of traditional package repositories. Used when imputing location information for packages not installed via `install_packages`
`...`	currently unused

Note

The manifest generated by this function will not include base packages, as they are part of R and not installable in the traditional sense.

Examples

```
man = libManifest()
man
## Not run:
man2 = libManifest("myotherlib")
man2

## End(Not run)
```

LibraryProfile-class *LibraryProfile (experimental)*

Description

Currently unused/under heavy development.

Details

An object

`library_paths` *library_paths*

Description

Accessor for which directories an SwitchrCtx is associated with.

Usage

```
library_paths(seed)

## S4 method for signature 'SwitchrCtx'
library_paths(seed)
```

Arguments

`seed`	An SwitchrCtx

`loadGRAN` *Load a GRAN repo package*

Description

Load a GRAN repo package

Usage

```
loadGRAN(nm = "current")
```

Arguments

`nm`	The name of the repository for which to load the package. Defaults `"current"`

Details

This function is a convenience to load the package GRAN<nm>, which will provide the contained GRAN repository as default repository within the switchr framework.

Value

NULL. Called for the side-effect of loading the specified package

`loadManifest` *loadManifest*

Description

Load a package or session manifest from a file (local or URL)

Usage

```
loadManifest(fil)
```

Arguments

`fil`	The path or URL to the file or a gist containing it

Value

A PkgManifest or SessionManifest object

`locatePkgVersion` *locatePkgVersion*

Description

Locate and download/build the exact version of a single package.

Usage

```
locatePkgVersion(name, version, pkg_manifest, param = SwitchrParam(),
  dir = notrack(repo), repo = NULL)
```

Arguments

`name`	package name
`version`	package version string
`pkg_manifest`	A manifest containing locations to search for the package
`param`	A SwitchrParam object
`dir`	directory to download package into
`repo`	(optional) GRANRepository object to search

Value

The full path to the downloaded file , or NULL if unable to locate the package

Note

Locating and attempting to install a non-current version of a single will not work in general, due to dependency issues. In most cases a Just-in-Time repository should be created and used instead, e.g. via `install_packages`

This function is called internally during the construction of Just-in-Time repositories and during the installation of specific package versions.

Author(s)

Gabriel Becker

`location` *location*

Description

Retreive the directory associated with an object

Usage

```
location(repo)

## S4 method for signature 'PkgSource'
location(repo)
```

Arguments

`repo`	An object associated with a path

Value

a character containing the associated path

Author(s)

Gabriel Becker

logfun *logfun*

Description

Get or set the logging function in an object associated with a SwitchrParam

Usage

```
logfun(x)

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

logfun(x) <- value

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

Arguments

x	An object with a SwitchrParam
value	The new logging function

makeBiocSVNURL *Make a Bioconductor SVN url for a package*

Description

Make SVN url for a Bioconductor package given the name, bioc version, and type of package.

Usage

```
makeBiocSVNURL(name, biocVers = getBiocvrFromRvr(),
  pkgtype = "software")
```

Arguments

name	A vector of bioconductor package names The name of the package
biocVers	The version (release) of bioconductor, or 'trunk' (the default) for Bioc devel.
pkgtype	character. Which type of packages to retrieve the SVN root url for. Should be "software" or "data" for software and experimental data packages, respectively.

Value

A vector of urls for the specified packages within the Bioconductor SVN repository

`makeFileURL` *make file url*

Description

make file url

Usage

```
makeFileURL(path)
```

Arguments

`path`	The path to wrap in a file:// URL

Value

A valid file URL

`makeLibraryCtx` *makeLibraryCtx*

Description

Locate or create a specified switchr library

Usage

```
makeLibraryCtx(name, seed = NULL, pkgs = NULL, exclude.site = TRUE,
  contains, rvers = NULL, verbose = FALSE)
```

Arguments

`name`	The name for the library
`seed`	The object to seed the library from
`pkgs`	Pkgs to install upon creation. Deprecated, use a seeding object instead.
`exclude.site`	Whether the site library should be excluded when switching to this library
`contains`	Currently unused.
`rvers`	Optional R version. If specified, existing libraries much be associated with the same R version to be considered a match.
`verbose`	Should informative messages be emitted to the console

Details

This function is not intended to be called directly in most cases; switchTo calls it automatically.

makeManifest *Manifest constructor*

Description

Create a package manifest

Usage

```
makeManifest(..., dep_repos = defaultRepos())
```

Arguments

`...` Vectors containing package information. Passed to `ManifestRow`

`dep_repos` The dependency repos for the package.

makePkgCheckout *Create a checkout of a package and all it's dependencies from a manifest*

Description

Create a checkout of a package and all it's dependencies from a manifest

Usage

```
makePkgCheckout(pkgs, pkg_manifest, dir, get_suggests = c("none",
  "first", "all"), param = SwitchrParam(),
  scm_auths = list(bioconductor = c("readonly", "readonly")),
  repos = defaultRepos())
```

Arguments

`pkgs` character - The packages you will be working on

`pkg_manifest` Pkgmanifest|SessionManifest - The manifest containing the pkgs and dependencies to checkout

`dir` character - The directory in which to place the checkouts of packages

`get_suggests` character - Should 'Suggests' dependencies be retrieved? Options are "none" (never), "first" (for packages in pkgs but not for dependencies, or "all" (always).

`param` SwitchrParam - The SwitchrParam to use during the checkout process,

`scm_auths` list - A named list of user-password pairs to use during the checkout process

`repos` character - The package repositories to retrieve dependnecy information from for pkgs/dependnecies which do not appear in `pkg_manifest`

Value

a character vector of all packages (incl. recursive dependnecies) checked out into `dir`

`makePkgDir` *makePkgDir*

Description

This is an internal function not intended to be called directly by end users

Usage

```
makePkgDir(name, source, path, latest_only, param = SwitchrParam(),
  forceRefresh = FALSE)

## S4 method for signature 'ANY,SVNSource'
makePkgDir(name, source, path,
  latest_only = FALSE, param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,GithubSource'
makePkgDir(name, source, path,
  latest_only = FALSE, param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,GitSource'
makePkgDir(name, source, path,
  latest_only = FALSE, param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,ANY'
makePkgDir(name, source, path, latest_only,
  param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,CRANSource'
makePkgDir(name, source, path, latest_only,
  param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,BiocSource'
makePkgDir(name, source, path, latest_only,
  param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,TarballSource'
makePkgDir(name, source, path, latest_only,
  param = SwitchrParam(), forceRefresh = FALSE)

## S4 method for signature 'ANY,LocalSource'
makePkgDir(name, source, path, latest_only,
  param = SwitchrParam(), forceRefresh = FALSE)
```

Arguments

`name` The package

`source` A PkgSource

`path` The path to place the directory

`latest_only` Should a fastpath for downloading the latest commit in a SCM package without a formal checkout be used?

`param` A SwitchrParam

`forceRefresh` Should an existing instance of the package source be deleted/refreshed

Details

Create a directory and populate it with package source code from the specified source

---

`makeSeedMan` *makeSeedMan*

Description

makeSeedMan

Usage

```
makeSeedMan(x, known_manifest = PkgManifest(), ...)

## S4 method for signature 'missing'
makeSeedMan(x, known_manifest = PkgManifest(), ...)

## S4 method for signature 'sessionInfo'
makeSeedMan(x, known_manifest = PkgManifest(),
  ...)

## S4 method for signature 'parsedSessionInfo'
makeSeedMan(x,
  known_manifest = PkgManifest(), ...)

## S4 method for signature 'data.frame'
makeSeedMan(x, known_manifest = PkgManifest(),
  ...)
```

Arguments

`x` The object to generate a seeding manifest from, if missing, the output from sessionInfo() is used.

`known_manifest` A manifest containing known locations of package sources. makeSeedMan will attempt to determine locations of packages listed in x using both known_manifest and official repositories.

`...` Currently unused.

Examples

```
man = makeSeedMan()
```

makeSource *Create a PkgSource object for a package*

Description

Create a PkgSource object for a package

Usage

```
makeSource(url, type, user, password, scm_auth = list(),
  prefer_svn = FALSE, ...)
```

Arguments

`url`	The url of the package sources
`type`	The source type.
`user`	A function which, when called, returns the username to use when when checking the soources out
`password`	A function which returns the password to use when checking out the sources
`scm_auth`	A list of username-password pairs, named with regular expressions to match against url when constructing the defaults for `user` and `password`
`prefer_svn`	Currently unused.
`...`	Passed directly to constructors for PkgSource superclasses

manifest *Get or set the manifest associated with an object*

Description

Get or set manifest associated with an object

Usage

```
manifest(x)

manifest(x) <- value

## S4 method for signature 'SessionManifest'
manifest(x)

## S4 replacement method for signature 'SessionManifest'
manifest(x) <- value
```

Arguments

`x`	An object which contains a manifest
`value`	A PkgManifest

Value

A PkgManifest or SessionManifest object

`ManifestRow` *ManifestRow*

Description

Create one or more rows of a manifest data.frame

Usage

```
ManifestRow(name, url = NA_character_, type = NA_character_,
  branch = NA_character_, subdir = ".", extra = NA_character_)
```

Arguments

`name`	name of the package.
`url`	location of the package sources
`type`	type of location (svn, git, local, etc)
`branch`	name of the branch to use to build the package
`subdir`	subdirectory to use to build the package
`extra`	currently ignored. extra commands for building or installing the package

Details

If name is missing, an empty (0 row) manifest data.frame is returned. All other fields default to values indicating no information- `NA_character` in most cases, and `"."` for `subdir`

Value

A valid Package manifest data.frame

manifest_df *manifest_df*

Description

Get or set the package location manifest (data.frame) associated with an object

Usage

```
manifest_df(x, ...)

## S4 method for signature 'SessionManifest'
manifest_df(x, session_only = TRUE, ...)

## S4 method for signature 'PkgManifest'
manifest_df(x)

manifest_df(x) <- value

## S4 replacement method for signature 'SessionManifest'
manifest_df(x) <- value

## S4 replacement method for signature 'PkgManifest'
manifest_df(x) <- value
```

Arguments

x	The object
...	unused.
session_only	Only return manifest rows associated with
value	A data.frame of package manifest information. See ManifestRow

normalizePath2 *normalizePath2*

Description

Attempt to normalize a relative path to an absolute one. Optionally without resolving symlinks on non-Windows systems

Usage

```
normalizePath2(path, follow.symlinks = FALSE, winslash = "\\",
  mustWork = NA)
```

Arguments

`path` The path to normalize

`follow.symlinks` Should symlinks (other than . and ..) be resolved to their physical locations? (FALSE)

`winslash` The value of winslash to be passed down to normalizePath on windows systems

`mustWork` logical. Passed to normalizePath on windows. Ignored otherwise.

Value

The normalized path.

`notrack` *Notrack directory*

Description

This function is not intended to be called directly by the user.

Usage

```
notrack(repo)

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

Arguments

`repo` The object.

Value

the path where retrieved package versions should be. If `repo` is NULL, a notrack directory is constructed within a temp directory.

nrow *Number of rows*

Description

Number of rows

Usage

```
nrow(x)

## S4 method for signature 'PkgManifest'
nrow(x)

## S4 method for signature 'SessionManifest'
nrow(x)
```

Arguments

x	A tabular data structure.

Value

The number of rows in the structure

packages *packages*

Description

List the packages installed in a switchr context (library)

Usage

```
packages(seed)

## S4 method for signature 'SwitchrCtx'
packages(seed)
```

Arguments

seed	A switchr context

parsedSessionInfo-class *Parsed sessionInfo output*

Description

An object representing the information in printed sessionInfo() output

parseSessionInfoString *Parse text output from printing SessionInfo objects*

Description

Parse text output from printing SessionInfo objects

Usage

```
parseSessionInfoString(string)
```

Arguments

`string`	The text output from sessionInfo()

PkgManifest *PkgManifest*

Description

Construct a PkgManifest, which can be installed from using `install_packages`

Usage

```
PkgManifest(manifest = ManifestRow(...), dep_repos = defaultRepos(),
  ..., dl_method)
```

Arguments

`manifest`	The manifest (data.frame) of packages and their locations
`dep_repos`	A list of traditional pkg repositories which can contain dependencies for the packages listed in `manifest`.
`...`	Arguments passed to `ManifestRow` if `manifest` is not specified
`dl_method`	Download method. Ignored unless `manifest` is a character scalar containing a URL to a serialized manifest

Details

If a package is found in both the manifest dataf.frame and the dependency repositories, the version in the manifest will always take precidence within the switchr framework.

pkgname *pkgname*

Description

Get or set the package name associated with a Package Source

Usage

```
pkgname(x)

## S4 method for signature 'PkgSource'
pkgname(x)

pkgname(x) <- value

## S4 replacement method for signature 'PkgSource'
pkgname(x) <- value
```

Arguments

x	A source
value	The new pkgname

PkgSource-class *PkgSource*

Description

An object representing the source location of a package. This is a virtual used exlusively through its subclasses, which are used to differentiate the different types of package source locations.

publishManifest *publishManifest*

Description

Publish a package or session manifest to file.

Usage

```
publishManifest(manifest, dest = "./pkg_manifest.rman", ...)

## S4 method for signature 'PkgManifest,character'
publishManifest(manifest,
  dest = "./pkg_manifest.rman", ...)

## S4 method for signature 'SessionManifest,character'
publishManifest(manifest,
  dest = "./pkg_manifest.rman", ...)

## S4 method for signature 'missing,ANY'
publishManifest(manifest,
  dest = "./pkg_manifest.rman", ...)

## S4 method for signature 'SwitchrCtx,ANY'
publishManifest(manifest,
  dest = "./pkg_manifest.rman", ...)
```

Arguments

`manifest`	The object to save as a serialized package or session manifest. Defaulst to the currently in use switchr library. A session manifest will be generated by libManifest as necessary.
`dest`	The destination `manifest` will be published to. Typically a character value indicating a file name (including path) to write to.
`...`	Unused

Value

The name of the file written

removeLib *removeLib*

Description

Remove a switchr library and update the manifest of existing libraries

Usage

```
removeLib(name = NULL, repos = NULL, compEnv = NULL,
  fromStack = FALSE)
```

Arguments

name	The name of the switchr library to remove
repos	the url used to seed the library
compEnv	a SwitchrCtx representing the library to remove
fromStack	Whether the library should be removed if it currently appears in the Context stack Defaults to false.

Value

NULL, called for its side-effect of removing/destroying a switchr library

Note

Only one of name, repos and compEnv should be specified. An error will be thrown otherwise.

Examples

```
## Not run:
removeLib("mylibrary")

## End(Not run)
```

RepoSubset *RepoSubset*

Description

An object that represents a subset of packages available in a repo. When switched to, switchr will default to only installing the specified packages, rather than all packages in the repository.

Usage

```
RepoSubset(repos, pkgs, default_name)
```

Arguments

`repos` The traditional repositories to select the packages from

`pkgs` The packages included in the subset

`default_name` The default name to use when the RepoSubset is used to seed a switchr context

`rVersionManifest` *rVersionManifest*

Description

Create a Pkg manifest which points to tarballs representing the cohort of packages associated with a particular release of R

Usage

```
rVersionManifest(vers, curr_avail = available.packages())
```

Arguments

`vers` The version of R to create a manifest for

`curr_avail` The output from available.packages(). Used to identify whether the necessary version is in the CRAN archive or normal repository

Value

A SessionManifest object

Author(s)

Gabriel Becker

References

"Gabor Csardi" (2014). crandb: Query the unofficial CRAN metadata database. R package version 1.0.0. https://github.com/metacran/crandb

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

Examples

```
## Not run:
man = rVersionManifest(”3.1.1”)
man

## End(Not run)
```

SessionManifest *SessionManifest*

Description

A manifest which includes both a PkgManifest containing package source information, and a data.frame defining a filter with exact versions of some or all packages

Usage

```
SessionManifest(manifest, versions = character())
```

Arguments

manifest	A PkgManifest
versions	A data.frame with 2 columns: name and version, or a named character vector. In the case of a character vector, the names are taken to be package names

Value

A SessionManifest object

shell_timing *Get or set the number of seconds to wait between successive shell commands*

Description

This is intended to stop intermittent install failures due to network drive latency interacting with git commands

Usage

```
shell_timing(x)

## S4 method for signature 'SwitchrParam'
shell_timing(x)

shell_timing(x) <- value

## S4 replacement method for signature 'SwitchrParam'
shell_timing(x) <- value
```

Arguments

`x`	A SwitchrParam object
`value`	The new number of seconds to wait

Value

When getting, the number of seconds to wait, when setting, a new, updated SwitchrParam object.

`sh_init_script` *shell init*

Description

Set or Retrieve the shell initialization script for an object

Usage

```
sh_init_script(x)

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

sh_init_script(x) <- value

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

Arguments

`x`	An object associated with a SwitchrParam object
`value`	The new value.

`subdir` *subdir*

Description

accessor for subdirectory.

Usage

```
subdir(x)

## S4 method for signature 'PkgSource'
subdir(x)

subdir(x) <- value

## S4 replacement method for signature 'PkgSource'
subdir(x) <- value
```

Arguments

`x`	An object associated with a subdirectory, typically a PkgSource
`value`	The new subdirectory to associate with the object

`switchBack` *switchBack*

Description

A convenience function to switch back to the previously used computing environment.

Usage

```
switchBack()
```

`switchDeps` *switchrDeps*

Description

The base packages, as well as switchr and its dependencies.

Usage

```
switchDeps
```

Format

An object of class `character` of length 20.

switchrBaseDir *Get or set the base directory for switchr libraries*

Description

Get or set the base directory for switchr libraries

Usage

```
switchrBaseDir(value)
```

Arguments

value	A new value for the base directory

Details

If value is missing, the current base directory is returned. Otherwise the value is set as the default directory and returned.

SwitchrCtx *SwitchrCtx*

Description

A constructor for class SwitchrCtx, represenging a switchr installed-package library.

Usage

```
SwitchrCtx(name, libpaths, exclude.site = TRUE, seed = NULL)
```

Arguments

name	The name to associate with the context
libpaths	The directories where the installed packages are located
exclude.site	Should the current site library be included in the context when it is switched to (TRUE) '
seed	An object representing the list of packages the switchr context was seeded with.

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

switchrDontUnload *Get or set packages to not unload when flushing the system*

Description

Get or set packages which should NOT be unloaded when flushing the system, e.g., when switching between libraries.

Usage

```
switchrDontUnload(value, add = TRUE)
```

Arguments

`value`	The packages to not unload when switching libraries.
`add`	Should `value` be added to the existing list?

Note

By default switchr will not attempt to unload any base packages, itself, or any of its dependencies. Attempting to unload any of these packages (e.g. `add=FALSE`) will result in undefined behavior and is not recommended.

switchrManifest *switchrManifest*

Description

Generate a manifest of all currently available (existing) swtichr libraries.

Usage

```
switchrManifest()
```

Value

A data.frame with information about the located switchr libraries

Note

This function reads cached metadata from the current switchr base directory (~/.switchr by default). This cache is updated whenever the switchr framework is used to create or destroy a switchr library, but will not be updated if one is added or removed manually. In such cases `updateManifest` must be called first

switchrNoUnload *Skip unloading of packages in session*

Description

Set whether or not ANY packages are unloaded when switching libraries.

Usage

```
switchrNoUnload(value)
```

Arguments

`value`	A logical value, or missing to return the current option

Details

This should be set to TRUE when using switchr in the context of dynamic documents such as .Rnw and .Rmd files.

Value

A logical indicating whether or not calling `flushSession` will skipped during the library switching process.

SwitchrParam-class *SwitchrParam*

Description

A contstructor for a SwitchrParam object representing a number of common parameters understood by the switchr framework

Usage

```
SwitchrParam(logfun = function(...) NULL, shell_init = character(),
  archive_timing = 2, archive_retries = 2, dl_method,
  shell_timing = 1)
```

Arguments

`logfun` The function to be called to write to logs

`shell_init` A character containing the location of a shell script to be sourced before any system commands.

`archive_timing` The timeout after downloading a package from the CRAN Archive.

`archive_retries` Number of times to retry retrieving a package from the CRAN Archive.

`dl_method` The download method to use when retrieve package source files. See `download.file` If none is specified, the method defaults to "curl" if the RCurl package is installed and "auto" otherwise.

`shell_timing` numeric. The numer of seconds to wait between certain shell commands. Defaults to 1, this should only need to be changed in the case of, e.g., networked drive latency issues.

Value

A SwitchrParam object.

Author(s)

Gabriel Becker

`switchTo` *switchTo*

Description

Switch to a different computing environment (set of installed R packages and library location paths for new pkg installs)

Usage

```
switchTo(name, seed = NULL, reverting = FALSE,
  ignoreRVersion = FALSE, exclude.site = TRUE, ...)

## S4 method for signature 'character,character'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)

## S4 method for signature 'character,SwitchrCtx'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)
```

```
## S4 method for signature 'character,missing'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)

## S4 method for signature 'SwitchrCtx,ANY'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)

## S4 method for signature 'character,RepoSubset'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)

## S4 method for signature 'character,PkgManifest'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)

## S4 method for signature 'character,SessionManifest'
switchTo(name, seed = NULL,
  reverting = FALSE, ignoreRVersion = FALSE, exclude.site = TRUE,
  ...)
```

Arguments

`name` The name associated (or to associate) with the computing environment.

`seed` The seed, indicating packages to install into a newly created package library No effect if the library already exists

`reverting` Indicates whether we are reverting to the environment in use before the current one. Typically not set directly by the user.

`ignoreRVersion` Should the R version in use be ignored when checking for existing computing environmeSnts. This is experimental.

`exclude.site` Should the Site library be excluded when creating and switching to the specified library. Defaults to `TRUE`

`...` Passed directly to `makeLibraryCtx` if an existing computing environment is not found.

Details

If switchr does not now about the specified computing environment, a new one will be created via installCompEnv. This includes creating a directory under the switchr base directory and installing packages into it. See `installCompEnv` for more details.

This function has the side effect of unloading all loaded packages (other than base packages, GRAN or GRANBAse, switchr itself, and switchr's dependencies) and the associated DLLs. It also changes the library location R will use to search for packages, e.g. when you call `library`.

This means you will have to reinstall packages after switching, which is important and intended (e.g. when switching to using Bioc devel from Bioc release).

Value

Invisibly returns the SwitchrCtx object representing the new computing environment

Note

By default, this process involves a call to `flushSession` which will attempt to unload all loaded packages. While some support of configuring what is unloaded is provided via `switchrDontUnload`, it is recommended that you turn this feature entirely off via `switchrNoUnload(TRUE)` when using switchr within dyanmic documents (.Rnw/.Rmd files, etc), particularly when using the knitr package.

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

See Also

`makeLibraryCtx`

Examples

```
## Not run:
switchTo("mynewlibrary")
switchBack()

fdman = GithubManifest("gmbecker/fastdigest")
switchTo("fastdigestlib", seed = fdman)

## End(Not run)
```

`system_w_init` *system_w_init*

Description

Run a system command with an optional intialization script (e.g. a .bashrc sourced first).

Usage

```
system_w_init(cmd, dir, init = character(), args = NULL, env = NULL,
  ..., param = SwitchrParam())
```

Arguments

`cmd` The text of the command. Must be length 1.

`dir` The directory that the command should be executed in. The working directory will be temporarily changed to this dir, but will be changed back upon exit of system_w_init.

`init` (optional) a character value indicating the location of an initialization shell script.

`args` character. Arguments to be passed to the command

`env` character. Environmental variables to be set when running the command

`...` additional parameters passed directly to `system`.

`param` A SwitchrParam object. The shell initialization script associated with this object is used when `init` is not specified (length 0).

Value

Depends, see `system` for details.

`updateManifest` *updateManifest*

Description

Update the cached information regarding available switchr libraries.

Usage

```
updateManifest()
```

Value

NULL, used for it's side-effect of updating the switchr library metadata cache.

`update_PACKAGES` *update existing package repository*

Description

Update an existing repository by reading the PACKAGES file and only processing built package tarballs which do not match existing entries.

`update_PACKAGES` can be much faster than `write_PACKAGES` for small-moderate changes to large repository indexes.

Usage

```
update_PACKAGES(dir = ".", fields = NULL, type = c("source",
  "mac.binary", "win.binary"), verbose = dryrun, unpacked = FALSE,
  subdirs = FALSE, latestOnly = TRUE, addFiles = FALSE,
  strict = TRUE, dryrun = FALSE, logfun = message, ...)
```

Arguments

`dir` See `write_PACKAGES`

`fields` See `write_PACKAGES`

`type` See `write_PACKAGES`

`verbose` Should informative messages be displayed throughout the proccess. Defaults to the value of `dryrun` (whose own default is `FALSE`) NOT passed to `write_PACKAGES`

`unpacked` See `write_PACKAGES`

`subdirs` See `write_PACKAGES`

`latestOnly` See `write_PACKAGES`

`addFiles` See `write_PACKAGES`

`strict` logical. Should 'strict mode' be used when checking existing PACKAGES entries. See details. Defaults to `TRUE`.

`dryrun` logical. Should should the necessary updates be calculated but NOT applied. (default `FALSE`)

`logfun` function. If `verbose` is `TRUE`, the function to be used to emit the informative messages. Defaults to `message`

`...` Additional arguments to `write_PACKAGES` - e.g., the relatively new rds_compress argument.

Details

Throughout this section, *package tarball* is taken to mean a tarball file in `dir` whose name can be interpreted as `<package>_<version>.<ext>` (or that is pointed to by the `File` field of an existing PACKAGES entry). *Novel package tarballs* are those which do not match an existing `PACKAGES` file entry.

`update_PACKAGES` avoids (re)processing package tarballs in cases where a `PACKAGES` file entry already exists and appears to remain valid. The logic for detecting still-valid entries is as follows:

Currently `update_PACKAGES` calls directly down to `write_PACKAGES` (and thus no speedup should be expected) if any of the following conditions hold:

- No `PACKAGES` file exists under `dir`
- `unpacked` is `TRUE`
- `subdirs` is anything other than `FALSE`
- `fields` is not `NULL` and one or more specified fields are not present in the existing `PACKAGES` file

All package tarballs whose last modify times are later than that of the existing PACKAGES file are considered novel and no attempt is made to identify or retain any corresponding PACKAGES entries. Similarly, all PACKAGES entries which have no corresponding package tarball are definitionally invalid.

When strict = TRUE, PACKAGES entries which appear to match a package tarball are confirmed via MD5 checksum; those that pass are retained as valid. All novel package tarballs are fully proccessed by the standard write_PACKAGES machinery, and the resulting entries are added. Finally, if latestOnly = TRUE, package-version pruning is performed across the entries.

When strict = FALSE, package tarballs are assumed to encode correct metadata in their filenames. PACKAGES entries which appear to match a package tarball are retained as valid (No MD5sum checking occurs). If latestOnly = TRUE, package-version pruning across the full set of retained entries and novel package tarballs *before* the processing of the novel tarballs, at significant computational and time savings in some situations. After the optional pruning, any relevant novel package tarballs are processed via write_PACKAGES and added to the set of retained entries.

After the above process concludes, the final database of PACKAGES entries is written to all three PACKAGES files, overwriting the existing files.

Note

While both strict and nonstrict modes offer speedups when updating small percentages of large repositories, non-strict mode is *much* faster and is recommended in situations where the assumptions it makes are safe.

Author(s)

Gabriel Becker

See Also

write_PACKAGES

---

versions_df *versions_df*

Description

Get or set the the versions information in a SessionManifest

Usage

```
versions_df(x)

## S4 method for signature 'SessionManifest'
versions_df(x)

versions_df(x) <- value
```

```
## S4 replacement method for signature 'SessionManifest'
versions_df(x) <- value
```

Arguments

x	An object containing package version information
value	A data.frame of package version information.

Index