Package: managelocalrepo (via r-universe)

June 7, 2024

Type Package

Title Manage a CRAN-Style Local Repository

Version 0.1.5

Date 2015-04-21

Author Imanuel Costigan <i.costigan@me.com>

Maintainer Imanuel Costigan <i.costigan@me.com>

Description This will allow easier management of a CRAN-style repository on local networks (i.e. not on CRAN). This might be necessary where hosted packages contain intellectual property owned by a corporation.

License GPL-2

Depends R (>= 3.0)

Imports stringr (>= 0.6.2), assertthat (>= 0.1), tools (>= 3.0)

Repository https://imanuelcostigan.r-universe.dev

RemoteUrl https://github.com/imanuelcostigan/managelocalrepo

RemoteRef HEAD

RemoteSha 8b87eeea6afae781f2f788298272cf91deec8549

Contents

create_terminal_dir *Create terminal repo directory*

Description

This will create the required terminal directory in the repo.

Usage

```
create_terminal_dir(repo_base, version, type = "win", ...)
```

Arguments

`repo_base`	the path of the base of the repository tree. This will have the following child folders: repo_base/bin/ and repo_base/src/. A character vector of length one.
`version`	the version of R that the package should be made available for. This is not relevant when `type` is `src`. A character vector of length one.
`type`	should be `"win"` (default), `"mac"` or `src` for Windows binary, Mac binary and source package distributions respectively.
`...`	optional arguments to pass to `file.path`

Value

No return value. Will create directory if the directory does not exist. Otherwise it will return an error message.

References

Setting up a package repository

Examples

```
## Not run:
repo_base <- file.path(".")
version <- '3.0'
create_terminal_dir(repo_base, version)

## End(Not run)
```

full_repo_dir *Determine the full path for local repo*

Description

This will return the full file path of the terminal directory of a given repo type (and R version number as is appropriate).

Usage

```
full_repo_dir(repo_base, version, type = "win", ...)
```

Arguments

`repo_base`	the path of the base of the repository tree. This will have the following child folders: repo_base/bin/ and repo_base/src/. A character vector of length one.
`version`	the version of R that the package should be made available for. This is not relevant when `type` is `src`. A character vector of length one.
`type`	should be `"win"` (default), `"mac"` or `src` for Windows binary, Mac binary and source package distributions respectively.
`...`	optional arguments to pass to `file.path`

Value

Full path to the desired local repo's terminal directory.

References

Setting up a package repository

Examples

```
## Not run:
repo_base <- file.path(".")
version <- '3.0'
full_repo_dir(repo_base, version)

## End(Not run)
```

managelocalrepo *Manage CRAN-like repo on local network*

Description

This package will allow easier maintainence of CRAN-like repos on local networks (i.e. not on CRAN). This might be necessary where hosted packages contain intellectual property owned by a corporation.

quick_release_package *Quick release a package*

Description

This is builds on top of release_package and makes it quicker to release packages. The repo base directory can be specified in the managelocalrepo.base option. The directory in which the package file is located is assumed to be a "submissions" folder and can be specified in the managelocalrepo.submissions option. These options can be set using options, potentially by using .First

Usage

```
quick_release_package(file_name, ...)
```

Arguments

file_name	the package's file name (not full path)
...	optional arguments to pass to release_package

Value

No return value. Will release a package to repo's terminal directory and update the index. Otherwise will return a suitable error message.

Examples

```
## Not run:
quick_release_package("test_package.zip")

## End(Not run)
```

release_package *Release a package to the local repo.*

Description

This will take a package from a given location and populate it to the relevant trees in the local repo. It will create terminal directories if needed.

Usage

```
release_package(package_location, repo_base, type = "win", version = "all",
  r_versions = c("2.15", "3.0", "3.1", "3.2"), ...)
```

Arguments

`package_location`
: the full path to the package's file. This should be the package's file name represented by a character vector of length one.

`repo_base`
: the base directory of the local repository represented by a character vector of length one.

`type`
: the type of package file being released. Should be `win` (default), `mac` or `src` for Windows binary, Mac binary and source package distributions respectively.

`version`
: determines which terminal directory to release the package file to, given a value for `type` that is either `wim` or `mac`. Can also use `"all"` to release the package to all terminal directories for either `wim` or `mac`. If `type` is `src`, then `version` is ignored, as this is not needed.

`r_versions`
: a character vector of R versions. This is used to determine the terminal directories when `type` is `"win"` or `"mac"` and `version` is `"all"`. Should follow the specifications set out in R Installation and Admin guide.

`...`
: optional arguments to pass to `file.path`

Value

No return value. Will release a package to the desired local repo's terminal directory and update the relevant index file. Otherwise will return a suitable error message.

References

Setting up a package repository

Examples

```
## Not run:
package_location <- file.path(".", "submissions")
package_location <- file.path(package_location, "test_package.zip")
repo_base <- file.path(".")
release_package(package_location, repo_base)

## End(Not run)
```

Index