Package 'projector'

June 6, 2020

Version 0.1.2

Date 2020-05-26

Title Projector Helps To Initiate And Develop R Packages

Description Projector is a more minimalistic and less extensive alternative to devtools/pkgbuilds/pkgload/rcmdcheck for initializing and developing R packages.

Depends R (>= 3.5.0)

Imports utils, methods

Suggests roxygen2, remotes

License GPL-3

URL https

LazyData true

Encoding UTF-8

RoxygenNote 6.1.1

R topics documented:

projector-package *projector: A short description of the package.*

Description

A more detailed explanation/description of the package

buildignore *Write .Rbuildignore*

Description

Write an .Rbuildignore file for a given package

Usage

```
buildignore(projname = ".", add = FALSE,
  pat = c("^commit\\.command$", "\\.Rproj$", "^__.*",
  "^\\.DS_Store$"), verbose = TRUE)
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`add`	logical; write new, or add to the existing file?
`pat`	a character vector of regex patterns
`verbose`	logical; print .Rbuildignore contents

desc_read *Read DESCRIPTION*

Description

Read the DESCRIPTION file of a given package

Usage

```
desc_read(projname = ".", quiet = FALSE)
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`quiet`	logical; if FALSE (default) results will be printed to console

See Also

`desc_write`, `increment_version`

`desc_write` *Write DESCRIPTION*

Description

Write a DESCRIPTION file for a given package

Usage

```
desc_write(projname = ".", descv)
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`descv`	a named character vector giving key-value pairs

See Also

`desc_read`, `increment_version`

`git_commit` *Git commit*

Description

Commit (and push) changes made to the git repository

Usage

```
git_commit(projname = ".", msg, add = ".", push = FALSE)
```

Arguments

`projname`	path to a folder with a git repository, by default "." (current directory)
`msg`	message to be included with the commit
`add`	what files to include, either as a single character string, or a character vector, where each element is a directory, file or fileglob.
`push`	logical; should the changes be pushed to a remote repository?

Details

This will add and commit, plus optionally push, all changes made in the given repository. Push assumes a very basic setup and pushes upstream to the master branch of 'origin'.

See Also

Other git functions: `git_init`, `git_push`, `git_remote`

git_init *Initialize git*

Description

Create an empty Git repository or reinitialize an existing one

Usage

```
git_init(projname = ".", add = ".", ignore = c(".DS_Store", ".RData",
  ".Rhistory", ".Rapp.history", ".Rproj.user"), commit = TRUE,
  msg = "first commit")
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`add`	what files to include, by default "." (all)
`ignore`	patterns to be written to .gitignore
`commit`	logical; should changes be commited?
`msg`	the message to be included with the commit

Details

An empty Git repository is created, files added, and a .gitignore file generated. Optionally the changes are commited and stored with a message.

See Also

Other git functions: `git_commit`, `git_push`, `git_remote`

git_push *Git push*

Description

Push changes to a remote repository

Usage

```
git_push(projname = ".")
```

Arguments

`projname`	path to a folder with a git repository, and a remote cennected. By default "." (current directory)

Details

This will push all changes made in the given local repository to the master branch of the 'origin' remote repository.

See Also

Other git functions: `git_commit`, `git_init`, `git_remote`

`git_remote` *Add git remote*

Description

Add a remote at a given github repository

Usage

```
git_remote(projname = ".", repo)
```

Arguments

`projname`	path to a folder with a git repository, by default "." (current directory)
`repo`	repository address in the format "username/repo[/subdir]"

Details

Connects a GitHub remote to the given git repository. This assumes a GitHub repository and SSH is already set up.

See Also

Other git functions: `git_commit`, `git_init`, `git_push`

`increment_version` *Increment version*

Description

Increment version number in the DESCRIPTION file of a given package

Usage

```
increment_version(projname = ".", comp = c("patch", "minor", "major"),
  dev = FALSE, quiet = FALSE)
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`comp`	which component to increment. By default "patch"
`dev`	logical; is it a development version?
`quiet`	logical; if FALSE (default) results will be printed to console

Details

Writes version numbers of the format "maj.min.patch.dev", but will also read formats with hyphen seperated components, like "maj.min-patch". If `dev` is `TRUE`, a version number of the form "maj.min.patch.9000" is used, otherwise "maj.min.patch". Other than version number, date, imports and R version number is updated as well.

See Also

`desc_read`, `desc_write`

`params` *Function parameters*

Description

Return the parameters of a function, formatted for Roxygen2 use

Usage

```
params(fun, to.clipboard = TRUE)
```

Arguments

`fun`	name of function
`to.clipboard`	logical; should results be copied to the clipboard?

Details

The parameters are returned as "`@param <par>`" pairs, one per line, as used by Roxygen2. The results can be either printed to console or copied to the clipboard, in either case a character vector will be returned invisibly as well.

See Also

`roxdoc`

Examples

```
params(params, FALSE)
params("params", FALSE)
```

pkg_check *Package check*

Description

Build and run diagnostic checks on the package

Usage

```
pkg_check(projname = ".", bopt = c("--no-manual"),
  copt = c("--no-manual", "--timings"), ropt = c("--vanilla"),
  rm.src = TRUE)
```

Arguments

`projname`	path to the project file
`bopt`	a character vector, or space-delimited character string, of options to be passed to `build`. By default `--no-manual`
`copt`	a character vector, or space-delimited character string, of options to be passed to `check`. By default `--vanilla` to run in a baseline R session, reading no profile or environment files, and no saving of workspace
`ropt`	a character vector, or space-delimited character string, of options to be passed to `R`. By default `--vanilla` to run in a baseline R session, reading no profile or environment files, and no saving of workspace
`rm.src`	remove source file

pkg_data *Prepare datasets*

Description

Save R objects as datasets

Usage

```
pkg_data(projname = ".", add = FALSE)
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`add`	logical; should datasets be added to those already existing in `data/`?

Details

All R objects in `__data.R` are saved to the folder `data/` as files on the pattern `d_<obj>.rda`, one for each object. If `add=FALSE`, then any existing files of that pattern will first be removed, before new ones are saved.

Value

.rda file(s) saved in the data/ folder.

See Also

pkg_objects

---

pkg_install *Install a package*

Description

Install a package

Usage

```
pkg_install(projname = ".", bopt = "", iopt = "",
  ropt = c("--vanilla"), rm.src = TRUE)
```

Arguments

projname	path to the project file
bopt	R CMD build options
iopt	R CMD install options
ropt	R options
rm.src	remove source file

---

pkg_objects *Object sizes*

Description

Find the size of a package's loadable objects

Usage

```
pkg_objects(projname = ".")
```

Arguments

projname	path to a package source, by default "." (current directory)

Details

The source code of a package is searched for function and dataset objects, found in the R/ and dataset/ directories.

Value

A data.frame giving the size, name, class, source file name and folder for each object

See Also

`pkg_data`

`pkg_pdf` *PDF manual*

Description

Render the PDF manual of all the package's documentation

Usage

```
pkg_pdf(projname = ".", ropt = "--vanilla", popt = "--force")
```

Arguments

`projname`	path to a package source, by default "." (current directory)
`ropt`	a character vector, or space-delimited character string, of options to be passed to `R`. By default `--vanilla` to run in a baseline R session, reading no profile or environment files, and no saving of workspace
`popt`	a character vector, or space-delimited character string, of options to be passed to `Rd2pdf`. By default `--force` to overwrite existing file

Value

A PDF file is saved to the root directory of the package

See Also

`Rd2pdf`

`roxdoc` *Documentation template*

Description

Create a function documentation template

Usage

```
roxdoc(fun, to.clipboard = TRUE)
```

Arguments

`fun`	function or name of function
`to.clipboard`	logical; should results be copied to the clipboard?

Details

The function arguments are returned as with `params`. In addition other commonly used Roxygen2 tags are included.

Value

Either to the console, or clipboard, a function documentation template, as appropriate for Roxygen2 use, is returned. Tags included are `@param` (with argument names), `@details`, `@return`, `@seealso`, `@examples` and `@export`.

See Also

`params`

Examples

```
roxdoc(roxdoc, FALSE)
roxdoc("roxdoc", FALSE)
```

`r_manual` *R manuals*

Description

Browse offline R manuals

Usage

```
r_manual(topic = c("extensions", "administration", "installation",
  "data", "I/O", "FAQ", "introduction", "internals",
  "language definition"))
```

Arguments

`topic`	which help topic to browse. By default "extensions" ("R-exts.html")

Details

The AQUA browser will be used if AQUA is used as the GUI, otherwise `getOption("browser")` is used.

Index