Package: pedev (via r-universe)

August 23, 2024

Type Package

Title Predictive Ecology Development Tools

Description Miscellaneous development tools developed by the Predictive Ecology Group (<https://predictiveecology.org>).

URL https://pedev.predictiveecology.org, https://github.com/PredictiveEcology/pedev

Date 2024-01-31

Version 0.0.1.9005

Depends R (>= 4.1)

Imports crayon, data.table, devtools, digest, igraph, methods, Require, reproducible, testthat, tools

Suggests covr, knitr, rmarkdown

Encoding UTF-8

Language en-CA

License GPL-3

VignetteBuilder knitr, rmarkdown

BugReports https://github.com/PredictiveEcology/pedev/issues

ByteCompile yes

LazyData true

RoxygenNote 7.3.1

Roxygen list(markdown = TRUE)

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

RemoteUrl https://github.com/PredictiveEcology/pedev

RemoteRef development

RemoteSha 1dbf81c73a4c1930716791916a278a0f5123298fd

Contents

`pedev-package` *pedev*

Description

Miscellaneous development tools developed by the Predictive Ecology Group (https://predictiveecology.org).

Author(s)

Maintainer: Eliot J B McIntire <eliot.mcintire@canada.ca> (ORCID)

Other contributors:

- Her Majesty the Queen in Right of Canada, as represented by the Minister of Natural Resources Canada [copyright holder]

See Also

Useful links:

- https://pedev.predictiveecology.org
- https://github.com/PredictiveEcology/pedev
- Report bugs at https://github.com/PredictiveEcology/pedev/issues

---

`file.sizeWOLinks` *Calculate file sizes omitting hard links*

Description

If two files are hardlinks, they don't actually take up extra space on disk: there is only one copy of the data and two pointers to the data.

Usage

```
file.sizeWOLinks(path = ".", units = "auto", recursive = FALSE)

file.sizeCompare(path = ".", units = "auto", recursive = TRUE)
```

Arguments

path	The path to evaluate file sizes in. Must be a directory.
units	the units to be used in formatting and printing the size. Allowed values for the different standards are standard = "legacy": "b", "Kb", "Mb", "Gb", "Tb", "Pb", "B", "KB", "MB", "GB", "TB" and "PB". standard = "IEC": "B", "KiB", "MiB", "GiB", "TiB", "PiB", "EiB", "ZiB" and "YiB". standard = "SI": "B", "kB", "MB", "GB", "TB", "PB", "EB", "ZB", "YB", "RB", and "QB". For all standards, unit = "auto" is also allowed. If standard = "auto", any of the "legacy" and IEC units are allowed. See 'Formatting and printing object sizes' for details.
recursive	Logical indicating whether to search recursively.

---

reload_all *A version of* devtools::load_all *that detaches dependencies*

Description

This is very idiosyncratic for the Predictive Ecology group.

Usage

```
reload_all(pkgs, load_all = TRUE, gitPath = "~/GitHub")
```

Arguments

pkgs	A character vector of the package(s) to run "devtools::load_all"
load_all	Logical. If FALSE, then this function will only detach the packages necessary.
gitPath	CHaracter giving the directory containing GitHub repos.

---

rmDotUnderline *Remove objects that start with "dot" "underscore"*

Description

A quick wrapper for removing a objects with specials names, "._xxx".

Usage

```
rmDotUnderline(envir = .GlobalEnv)
```

Arguments

envir	The environment from with to rm objects, defaults to .GlobalEnv

`updateGit` *Pull from git and install*

Description

Fetches all branches, then pulls the identified branch from git, then runs a digest on the local folders. If that digest is different as a previous one, then the function will run `devtools::install(dependencies = FALSE, reload = FALSE, quick = TRUE, ...)`. This should be safe even in cases where local files have changed. If they were uncommitted, Git will error, and nothing will be pulled, and if they were committed, then it will try a merge. If the automoated merge works, then it will proceed. If automated merge fails, then nothing will be pulled.

Usage

```
updateGit(
  pkgs = NULL,
  install = TRUE,
  branch = c("development", "master"),
  cacheRepo = getOption("pedev.cacheRepo", "~/.pedevCache"),
  fetch = TRUE,
  submodule = FALSE,
  quick = TRUE,
  dependencies = FALSE,
  reload = FALSE,
  ...
)
```

Arguments

`pkgs`	A character vector of package names, which is actually the path names of the packages. i.e., must be absolute or relative path. Defaults to current directory. It will also check in "..", i.e., one folder up from the current active folder if it doesn't find `pkgs` in the current folder.
`install`	Logical. If TRUE, then it will run `devtools::install` if there is new content. If `branch` was length > 1, only the active, i.e., first branch, will be installed.
`branch`	A vector of branch names to pull from, *in reverse order* so that the first one is the active branch after this function call finishes. Default is `c("development", "master")`, so it will pull from master, then development. If one of them does not exist, it will try, deteremine it doesn't exist, skip it and go to next branch.
`cacheRepo`	The location where subsequent calls will store their history. To be most effective, this should be "persistent", and not part of any other cacheRepo.
`fetch`	Logical. Should it fetch before pulling.
`submodule`	Logical. VERY EXPERIMENTAL. `TRUE` would mean pull all submodules... the problem is that branch gets complicated.
`quick`	if `TRUE` skips docs, multiple-architectures, demos, and vignettes, to make installation as fast as possible.

dependencies	Which dependencies do you want to check? Can be a character vector (selecting from "Depends", "Imports", "LinkingTo", "Suggests", or "Enhances"), or a logical vector. TRUE is shorthand for "Depends", "Imports", "LinkingTo" and "Suggests". NA is shorthand for "Depends", "Imports" and "LinkingTo" and is the default. FALSE is shorthand for no dependencies (i.e. just check this package, not its dependencies). The value "soft" means the same as TRUE, "hard" means the same as NA. You can also specify dependencies from one or more additional fields, common ones include: • Config/Needs/website - for dependencies used in building the pkgdown site. • Config/Needs/coverage for dependencies used in calculating test coverage.
reload	if TRUE (the default), will automatically reload the package after installing.
...	Passed to devtools::install

Examples

```
## Not run:
# This will pull development branch of all these packages, and install them
#    all, if there are any file changes in each respective directory
allPkgs <- c("quickPlot", "reproducible", "SpaDES.core", "SpaDES.tools",
             "pemisc", "map", "LandR", "pedev")
updateGit(allPkgs)

# Will update and install all development branches of all repositories
#   in ~/GitHub folder
pedev::updateGit(dir("~/GitHub"))


## End(Not run)
```

Index