Package ‘githubinstall’

November 21, 2016

Type Package
Version 0.2.1
Title A Helpful Way to Install R Packages Hosted on GitHub
Description Provides an helpful way to install packages hosted on GitHub.
URL https://github.com/hoxo-m/githubinstall
BugReports https://github.com/hoxo-m/githubinstall/issues
License MIT + file LICENSE
Imports curl, data.table, devtools, httr, jsonlite, utils
RoxygenNote 5.0.1
Suggests testthat, ggplot2, lubridate, knitr, rmarkdown
VignetteBuilder knitr
NeedsCompilation no
Author Koji Makiyama [cre, aut],
Atsushi Hayakawa [ctb],
Shinya Uryu [ctb],
Hiroaki Yutani [ctb],
Nagi Teramo [ctb]
Maintainer Koji Makiyama <hoxo.smile@gmail.com>
Repository CRAN
Date/Publication 2016-11-21 12:01:55

R topics documented:

<table>
<thead>
<tr>
<th>R function</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>gh_install_packages</td>
<td>2</td>
</tr>
<tr>
<td>gh_list_packages</td>
<td>3</td>
</tr>
<tr>
<td>gh_search_packages</td>
<td>4</td>
</tr>
<tr>
<td>gh_show_source</td>
<td>4</td>
</tr>
<tr>
<td>gh_suggest</td>
<td>5</td>
</tr>
<tr>
<td>gh_suggest_username</td>
<td>6</td>
</tr>
<tr>
<td>gh_update_package_list</td>
<td>6</td>
</tr>
</tbody>
</table>
Install Packages from GitHub

### Usage

```r
gh_install_packages(packages, ask = TRUE, ref = "master",
build_vignettes = FALSE, dependencies = NA, verbose = TRUE,
quiet = !verbose, lib = NULL, ...)
```

```r
githubinstall(packages, ask = TRUE, ref = "master",
build_vignettes = FALSE, dependencies = NA, verbose = TRUE,
quiet = !verbose, lib = NULL, ...)
```

### Arguments

- **packages** character vector of the names of the packages. You can specify ref argument (see below) using `package_name[@ref=#pull]`. If both are specified, the values in repo take precedence.
- **ask** logical. Indicates ask to confirm before install.
- **ref** character vector. Desired git reference. Could be a commit, tag, or branch name, or a call to `github_pull`. Defaults to "master".
- **build_vignettes** logical. If TRUE, will build vignettes.
- **dependencies** logical. Indicating to also install uninstalled packages which the packages depends on/links to/suggests. See argument dependencies of `install.packages`.
- **verbose** logical. Indicating to print details of package building and installation. Default is TRUE.
- **quiet** logical. Not verbose.
- **lib** character vector giving the library directories where to install the packages. Recycled as needed. Defaults to the first element of `.libPaths()`.
- **...** additional arguments to control installation of package, passed to `install_github`.

### Details

`githubinstall()` is an alias of `gh_install_packages()`.

---

```r
recommend_dependencies ........................................ 7
remove_conflict_repos ........................................ 7
select_repository .............................................. 8
separate_into_package_and_reference ...................... 8
```
gh_list_packages

Value

TRUE if success.

Examples

## not run:
gh_install_packages("AnomalyDetection")
githubinstall("AnomalyDetection")

## End(Not run)

gh_list_packages  Get Information of Packages on GitHub

Description

Get Information of Packages on GitHub

Usage

gh_list_packages(username = NULL)

Arguments

username a character vector as GitHub username. If you set NULL (default), it returns all packages information.

Value

a data.frame that has author, package name and title.

Examples

## Not run:
gh_list_packages("hadley")

## End(Not run)
### gh_search_packages

**Search Packages from Titles by Regular Expressions**

**Description**

Search Packages from Titles by Regular Expressions

**Usage**

`gh_search_packages(regex, ignore.case = TRUE)`

**Arguments**

- `regex`: a character string containing a regular expression to be matched in the package titles.
- `ignore.case`: logical. If FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

**Value**

a data.frame of package information.

**Examples**

```r
## not run:
gh_search_packages("lasso")
## end(not run)
```

---

### gh_show_source

**Find source code for functions in packages on GitHub**

**Description**

Find source code for functions in packages on GitHub

**Usage**

`gh_show_source(func, repo = NULL, browser = getOption("browser"))`

**Arguments**

- `func`: a function or a character string. A function name.
- `repo`: a character string. A GitHub repository name that must not be exactry.
- `browser`: a character string giving the name of the program to be used as the HTML browser.
### gh_suggest

**Examples**

```r
## Not run:
gh_show_source("mutate", "dplyr")

library(dplyr)
gh_show_source(mutate)

## End(Not run)
```

---

**Description**

Suggest Github Repository from a Incomplete Name

**Usage**

```r
gh_suggest(repo_name, keep_title = FALSE)
```

**Arguments**

- `repo_name` a character. A part of a repository name.
- `keep_title` logical. Indicates to keep the package titles as an attribute. Default FALSE.

**Value**

candidates for the repository name.

**Examples**

```r
gh_suggest("AnomalyDetection")
# [1] "twitter/AnomalyDetection"
gh_suggest("BnomalyDetection")
# [1] "twitter/AnomalyDetection"
gh_suggest("uwitter/BnomalyDetection")
# [1] "twitter/AnomalyDetection"
```
gh_suggest_username  Suggest Github Username from a Faint Memory

Description
Suggest Github Username from a Faint Memory

Usage
gh_suggest_username(vague_name)

Arguments
vague_name  a character. GitHub username that may not be exact.

Details
The trouble is that the usernames of GitHub are often hard to remember. The function provides a way to obtain usernames from a faint memory.

Value
a character vector of the closest usernames to input.

Examples
## not run:
gh_guess_username(ByuhuiB)
# [1] ByihuiB
## end(not run)

gh_update_package_list
Update the List of Packages on GitHub.

Description
Update the List of Packages on GitHub.

Usage
gh_update_package_list()
**recommend_dependencies**

The default "dependencies" is NA that means c("Depends", "Imports", "LinkingTo"). If "build_vignettes" is TRUE, the install needs "Suggests" dependency in many cases. So we recommend in such case to set "dependencies" to TRUE that means c("Depends", "Imports", "LinkingTo", "Suggests").

**Description**

The default "dependencies" is NA that means c("Depends", "Imports", "LinkingTo"). If "build_vignettes" is TRUE, the install needs "Suggests" dependency in many cases. So we recommend in such case to set "dependencies" to TRUE that means c("Depends", "Imports", "LinkingTo", "Suggests").

**Usage**

```r
recommend_dependencies(ask, build_vignettes, dependencies, quiet)
```

**Arguments**

- **ask** logical. Indicates ask to confirm before install.
- **build_vignettes** logical. If TRUE, will build vignettes.
- **dependencies** logical. Indicating to also install uninstalled packages which the packages depends on/links to/suggests. See argument dependencies of `install.packages`.
- **quiet** logical. Not verbose.

**remove_conflict_repos**

We want to detect the two conflict cases as follows: 1. The package is already installed from some repository like CRAN that is not GitHub. 2. The package is already installed from GitHub but the username differs. In the above cases, we ask whether to overwrite it and remove from "repo" if the answer is no.

**Description**

If "quiet" is TRUE, we overwrite all packages forcibly and silently. Else if "quiet" is FALSE and "ask" is TRUE, we ask whether to overwrite it. (Default) Else if "quiet" is FALSE and "ask" is FALSE, we message to overwrite it and do it.

**Usage**

```r
remove_conflict_repos(repos, lib, quiet, ask)
```
Arguments
repos  character vector of full GitHub repository names.
lib    character vector or NULL.
quiet  logical.
ask    logical.

select_repository  Suggest candidates from "package_name" and make user selected one of them.

Description
Suggest candidates from "package_name" and make user selected one of them.

Usage
select_repository(package_name)

Arguments
package_name  a character string. A package name or full GitHub repository name.

Value
candidate with title

separate_into_package_and_reference
The "repo" argument allows to contain "ref" as "package_name@ref", "package_name#pull" or "package_name[branch]". The function detects that "repo" contains "ref" and separates into pure repo and ref. If "repo" contains "ref" and "ref" argument is specified, the values in "repo" take precedence.

Description
The "repo" argument allows to contain "ref" as "package_name@ref", "package_name#pull" or "package_name[branch]". The function detects that "repo" contains "ref" and separates into pure repo and ref. If "repo" contains "ref" and "ref" argument is specified, the values in "repo" take precedence.

Usage
separate_into_package_and_reference(packages, original_ref)
Arguments

- packages: "repo" argument.
- original_ref: "ref" argument.
Index

.libPaths, 2

gh_install_packages, 2
gh_list_packages, 3
gh_search_packages, 4
gh_show_source, 4
gh_suggest, 5
gh_suggest_username, 6
gh_update_package_list, 6
github_pull, 2
githubinstall (gh_install_packages), 2

install.packages, 2, 7
install_github, 2

recommend_dependencies, 7
regular expression, 4
remove_conflict_repos, 7

select_repository, 8
separate_into_package_and_reference, 8