Package 'remotes'

July 23, 2025

Title R Package Installation from Remote Repositories, Including 'GitHub'

Version 2.5.0

Description Download and install R packages stored in 'GitHub', 'GitLab', 'Bitbucket', 'Bioconductor', or plain 'subversion' or 'git' repositories. This package provides the 'install_*' functions in 'devtools'. Indeed most of the code was copied over from 'devtools'.

License MIT + file LICENSE

URL https://remotes.r-lib.org, https://github.com/r-lib/remotes#readme

BugReports https://github.com/r-lib/remotes/issues

Depends R (>= 3.0.0)

Imports methods, stats, tools, utils

Suggests brew, callr, codetools, covr, curl, git2r (>= 0.23.0), knitr, mockery, pingr, pkgbuild (>= 1.0.1), rmarkdown, rprojroot, testthat (>= 3.0.0), webfakes, withr

VignetteBuilder knitr

Config/Needs/website tidyverse/tidytemplate

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.3

SystemRequirements Subversion for install_svn, git for install_git

NeedsCompilation no

Author Gábor Csárdi [aut, cre],
Jim Hester [aut],
Hadley Wickham [aut],
Winston Chang [aut],
Martin Morgan [aut],
Dan Tenenbaum [aut],
Posit Software, PBC [cph, fnd],
Ascent Digital Services [cph]

Maintainer Gábor Csárdi <csardi.gabor@gmail.com>

Repository CRAN

Date/Publication 2024-03-17 13:20:02 UTC

Contents

`download_version` *Download a specified version of a CRAN package*

Description

It downloads the package to a temporary file, and returns the name of the file.

Usage

```
download_version(
  package,
  version = NULL,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

package	Name of the package to install.
version	Version of the package to install. Can either be a string giving the exact version required, or a specification in the same format as the parenthesized expressions used in package dependencies. One of the following formats: • An exact version required, as a string, e.g. `"0.1.13"` • A comparison operator and a version, e.g. `">= 0.1.12"` • Several criteria to satisfy, as a comma-separated string, e.g. `">= 1.12.0, < 1.14"` • Several criteria to satisfy, as elements of a character vector, e.g. `c(">= 1.12.0", "< 1.14")`
repos	character vector, the base URL(s) of the repositories to use, e.g., the URL of a CRAN mirror such as `"https://cloud.r-project.org"`. For more details on supported URL schemes see `url`. Can be `NULL` to install from local files, directories or URLs: this will be inferred by extension from `pkgs` if of length one.
type	character, indicating the type of package to download and install. Will be `"source"` except on Windows and some macOS builds: see the section on 'Binary packages' for those.
...	Other arguments passed on to `utils::install.packages()`.

Value

Name of the downloaded file.

`github_pull` *GitHub references*

Description

Use as `ref` parameter to `install_github()`. Allows installing a specific pull request or the latest release.

Usage

```
github_pull(pull)

github_release()
```

Arguments

pull	Character string specifying the pull request to install

See Also

`install_github()`

Examples

```
github_pull("42")
```

`install_bioc` *Install a development package from the Bioconductor git repository*

Description

This function requires `git` to be installed on your system in order to be used.

Usage

```
install_bioc(
  repo,
  mirror = getOption("BioC_git", download_url("git.bioconductor.org/packages")),
  git = c("auto", "git2r", "external"),
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

`repo`	Repository address in the format `[username:password@][release/]repo[#commit]`. Valid values for the release are 'devel', 'release' (the default if none specified), or numeric release numbers (e.g. '3.3').
`mirror`	The Bioconductor git mirror to use
`git`	Whether to use the `git2r` package, or an external git client via system. Default is `git2r` if it is installed, otherwise an external git installation.
`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.

`upgrade` Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the `R_REMOTES_UPGRADE` environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". `TRUE` and `FALSE` are also accepted and correspond to "always" and "never" respectively.

`force` Force installation, even if the remote state has not changed since the previous install.

`quiet` If `TRUE`, suppress output.

`build` If `TRUE` build the package before installing.

`build_opts` Options to pass to `R CMD build`, only used when `build` is `TRUE`.

`build_manual` If `FALSE`, don't build PDF manual ('–no-manual').

`build_vignettes` If `FALSE`, don't build package vignettes ('–no-build-vignettes').

`repos` A character vector giving repositories to use.

`type` Type of package to `update`.

`...` Other arguments passed on to `utils::install.packages()`.

Details

It is vectorised so you can install multiple packages with a single command.

This is intended as an aid for Bioconductor developers. If you want to install the release version of a Bioconductor package one can use the `BiocManager` package.

See Also

Other package installation: `install_bitbucket()`, `install_cran()`, `install_dev()`, `install_github()`, `install_gitlab()`, `install_git()`, `install_local()`, `install_svn()`, `install_url()`, `install_version()`

Examples

```
## Not run:
install_bioc("SummarizedExperiment")
install_bioc("devel/SummarizedExperiment")
install_bioc("3.3/SummarizedExperiment")
install_bioc("SummarizedExperiment#abc123")
install_bioc("user:password@release/SummarizedExperiment")
install_bioc("user:password@devel/SummarizedExperiment")
install_bioc("user:password@SummarizedExperiment#abc123")

## End(Not run)
```

`install_bitbucket` *Install a package directly from Bitbucket*

Description

This function is vectorised so you can install multiple packages in a single command.

Usage

```
install_bitbucket(
  repo,
  ref = "HEAD",
  subdir = NULL,
  auth_user = bitbucket_user(),
  password = bitbucket_password(),
  host = "api.bitbucket.org/2.0",
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

`repo`	Repository address in the format `username/repo[/subdir][@ref]`. Alternatively, you can specify `subdir` and/or `ref` using the respective parameters (see below); if both are specified, the values in `repo` take precedence.
`ref`	Desired git reference; could be a commit, tag, or branch name. Defaults to HEAD.
`subdir`	Subdirectory within repo that contains the R package.
`auth_user`	your account username if you're attempting to install a package hosted in a private repository (and your username is different to `username`). Defaults to the `BITBUCKET_USER` environment variable.
`password`	your password. Defaults to the `BITBUCKET_PASSWORD` environment variable. See details for further information on setting up a password.
`host`	GitHub API host to use. Override with your GitHub enterprise hostname, for example, `"github.hostname.com/api/v3"`.

`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.

`upgrade` Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the `R_REMOTES_UPGRADE` environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". `TRUE` and `FALSE` are also accepted and correspond to "always" and "never" respectively.

`force` Force installation, even if the remote state has not changed since the previous install.

`quiet` If `TRUE`, suppress output.

`build` If `TRUE` build the package before installing.

`build_opts` Options to pass to `R CMD build`, only used when `build` is `TRUE`.

`build_manual` If `FALSE`, don't build PDF manual ('–no-manual').

`build_vignettes`
If `FALSE`, don't build package vignettes ('–no-build-vignettes').

`repos` A character vector giving repositories to use.

`type` Type of package to `update`.

`...` Other arguments passed on to `utils::install.packages()`.

Details

To install from a private repo, or more generally, access the Bitbucket API with your own credentials, you will need to get an access token. You can create an access token following the instructions found in the Bitbucket App Passwords documentation. The App Password requires read-only access to your repositories and pull requests. Then store your password in the environment variable `BITBUCKET_PASSWORD` (e.g. `evelynwaugh:swordofhonour`)

Note that on Windows, authentication requires the "libcurl" download method. You can set the default download method via the `download.file.method` option:

```
options(download.file.method = "libcurl")
```

In particular, if unset, RStudio sets the download method to "wininet". To override this, you might want to set it to "libcurl" in your R profile, see base::Startup. The caveat of the "libcurl" method is that it does *not* set the system proxies automatically, see "Setting Proxies" in `utils::download.file()`.

See Also

Bitbucket API docs: https://confluence.atlassian.com/bitbucket/use-the-bitbucket-cloud-rest-apis-22272 html

Other package installation: install_bioc(), install_cran(), install_dev(), install_github(), install_gitlab(), install_git(), install_local(), install_svn(), install_url(), install_version()

Examples

```
## Not run:
install_bitbucket("sulab/mygene.r@default")
install_bitbucket("djnavarro/lsr")

## End(Not run)
```

---

install_cran *Attempts to install a package from CRAN.*

Description

This function is vectorised on pkgs so you can install multiple packages in a single command.

Usage

```
install_cran(
  pkgs,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  ...
)
```

Arguments

pkgs	A character vector of packages to install.
repos	A character vector giving repositories to use.
type	Type of package to update.

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.

upgrade
: Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.

force
: Force installation, even if the remote state has not changed since the previous install.

quiet
: If TRUE, suppress output.

build
: If TRUE build the package before installing.

build_opts
: Options to pass to R CMD build, only used when build is TRUE.

build_manual
: If FALSE, don't build PDF manual ('–no-manual').

build_vignettes
: If FALSE, don't build package vignettes ('–no-build-vignettes').

...
: Other arguments passed on to utils::install.packages().

See Also

Other package installation: install_bioc(), install_bitbucket(), install_dev(), install_github(), install_gitlab(), install_git(), install_local(), install_svn(), install_url(), install_version()

Examples

```
## Not run:
install_cran("ggplot2")
install_cran(c("httpuv", "shiny"))

## End(Not run)
```

`install_deps` *Install package dependencies if needed.*

Description

Install package dependencies if needed.

Usage

```
install_deps(
  pkgdir = ".",
  dependencies = NA,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  upgrade = c("default", "ask", "always", "never"),
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  ...
)
```

Arguments

`pkgdir`	Path to a package directory, or to a package tarball.
`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.
`repos`	A character vector giving repositories to use.
`type`	Type of package to `update`.
`upgrade`	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the `R_REMOTES_UPGRADE` environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". `TRUE` and `FALSE` are also accepted and correspond to "always" and "never" respectively.

quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
...	additional arguments passed to utils::install.packages().

Examples

```
## Not run: install_deps(".")
```

---

install_dev *Install the development version of a package*

Description

install_dev() retrieves the package DESCRIPTION from the CRAN mirror and looks in the 'URL' and 'BugReports' fields for GitHub, GitLab or Bitbucket URLs. It then calls the appropriate install_() function to install the development package.

Usage

```
install_dev(package, cran_url = getOption("repos")[["CRAN"]], ...)
```

Arguments

package	The package name to install.
cran_url	The URL of the CRAN mirror to use, by default based on the 'repos' option. If unset uses 'https://cloud.r-project.org'.
...	Additional arguments passed to install_github(), install_gitlab(), or install_bitbucket() functions.

See Also

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_github(), install_gitlab(), install_git(), install_local(), install_svn(), install_url(), install_version()

Examples

```
## Not run:
# From GitHub
install_dev("dplyr")

# From GitLab
install_dev("iemiscdata")
```

```
# From Bitbucket
install_dev("argparser")

## End(Not run)
```

`install_git` *Install a package from a git repository*

Description

It is vectorised so you can install multiple packages with a single command. You do not need to have the `git2r` package, or an external git client installed.

Usage

```
install_git(
  url,
  subdir = NULL,
  ref = NULL,
  branch = NULL,
  credentials = git_credentials(),
  git = c("auto", "git2r", "external"),
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

`url`	Location of package. The url should point to a public or private repository.
`subdir`	A sub-directory within a git repository that may contain the package we are interested in installing.
`ref`	Name of branch, tag or SHA reference to use, if not HEAD.
`branch`	Deprecated, synonym for ref.
`credentials`	A git2r credentials object passed through to clone. Supplying this argument implies using `git2r` with `git`.

`git` Whether to use the `git2r` package, or an external git client via system. Default is `git2r` if it is installed, otherwise an external git installation.

`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.

`upgrade` Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the `R_REMOTES_UPGRADE` environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". `TRUE` and `FALSE` are also accepted and correspond to "always" and "never" respectively.

`force` Force installation, even if the remote state has not changed since the previous install.

`quiet` If `TRUE`, suppress output.

`build` If `TRUE` build the package before installing.

`build_opts` Options to pass to `R CMD build`, only used when `build` is `TRUE`.

`build_manual` If `FALSE`, don't build PDF manual ('–no-manual').

`build_vignettes`
If `FALSE`, don't build package vignettes ('–no-build-vignettes').

`repos` A character vector giving repositories to use.

`type` Type of package to `update`.

`...` Other arguments passed on to `utils::install.packages()`.

Details

If you need to set git credentials for use in the `Remotes` field you can do so by placing the credentials in the `remotes.git_credentials` global option.

See Also

Other package installation: `install_bioc()`, `install_bitbucket()`, `install_cran()`, `install_dev()`, `install_github()`, `install_gitlab()`, `install_local()`, `install_svn()`, `install_url()`, `install_version()`

Examples

```
## Not run:
install_git("https://github.com/hadley/stringr.git")
install_git("https://github.com/hadley/stringr.git", ref = "stringr-0.2")

## End(Not run)
```

`install_github` *Attempts to install a package directly from GitHub.*

Description

This function is vectorised on `repo` so you can install multiple packages in a single command.

Usage

```
install_github(
  repo,
  ref = "HEAD",
  subdir = NULL,
  auth_token = github_pat(quiet),
  host = "api.github.com",
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

`repo`	Repository address in the format `username/repo[/subdir][@ref|#pull|@*release]`. Alternatively, you can specify `subdir` and/or `ref` using the respective parameters (see below); if both are specified, the values in `repo` take precedence.
`ref`	Desired git reference. Could be a commit, tag, or branch name, or a call to `github_pull()` or `github_release()`. Defaults to `"HEAD"`, which means the default branch on GitHub and for git remotes. See setting-the-default-branch for more details.
`subdir`	Subdirectory within repo that contains the R package.

auth_token	To install from a private repo, generate a personal access token (PAT) with at least repo scope in https://github.com/settings/tokens and supply to this argument. This is safer than using a password because you can easily delete a PAT without affecting any others. Defaults to the GITHUB_PAT environment variable.
host	GitHub API host to use. Override with your GitHub enterprise hostname, for example, "github.hostname.com/api/v3".
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.
upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Force installation, even if the remote state has not changed since the previous install.
quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
repos	A character vector giving repositories to use.
type	Type of package to update.
...	Other arguments passed on to utils::install.packages().

Details

If the repository uses submodules a command-line git client is required to clone the submodules.

See Also

github_pull()

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_dev(), install_gitlab(), install_git(), install_local(), install_svn(), install_url(), install_version()

Examples

```
## Not run:
install_github("klutometis/roxygen")
install_github("wch/ggplot2", ref = github_pull("142"))
install_github(c("rstudio/httpuv", "rstudio/shiny"))
install_github(c("hadley/httr@v0.4", "klutometis/roxygen#142",
  "r-lib/roxygen2@*release", "mfrasca/r-logging/pkg"))

# To install from a private repo, use auth_token with a token
# from https://github.com/settings/tokens. You only need the
# repo scope. Best practice is to save your PAT in env var called
# GITHUB_PAT.
install_github("hadley/private", auth_token = "abc")

# To pass option arguments to `R CMD INSTALL` use `INSTALL_opts`. e.g. to
install a package with source references and tests
install_github("rstudio/shiny", INSTALL_opts = c("--with-keep.source", "--install-tests"))

## End(Not run)
```

---

install_gitlab *Install a package from GitLab*

Description

This function is vectorised on `repo` so you can install multiple packages in a single command. Like other remotes the repository will skip installation if `force == FALSE` (the default) and the remote state has not changed since the previous installation.

Usage

```
install_gitlab(
  repo,
  subdir = NULL,
  auth_token = gitlab_pat(quiet),
  host = "gitlab.com",
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

repo	Repository address in the format username/repo[@ref].
subdir	Subdirectory within repo that contains the R package.
auth_token	To install from a private repo, generate a personal access token (PAT) with at least read_api scope in https://docs.gitlab.com/ee/user/profile/personal_access_tokens.html and supply to this argument. This is safer than using a password because you can easily delete a PAT without affecting any others. Defaults to the GITLAB_PAT environment variable.
host	GitLab API host to use. Override with your GitLab enterprise hostname, for example, "<PROTOCOL://>gitlab.hostname.com". The PROTOCOL is required by packrat during Posit Connect deployment. While install_gitlab may work without, omitting it generally leads to package restoration errors.
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.
upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Force installation, even if the remote state has not changed since the previous install.
quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
repos	A character vector giving repositories to use.
type	Type of package to update.
...	Other arguments passed on to utils::install.packages().

See Also

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_dev(), install_github(), install_git(), install_local(), install_svn(), install_url(), install_version()

Examples

```
## Not run:
install_gitlab("jimhester/covr")

## End(Not run)
```

`install_local` *Install a package from a local file*

Description

This function is vectorised so you can install multiple packages in a single command.

Usage

```
install_local(
  path = ".",
  subdir = NULL,
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = !is_binary_pkg(path),
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

path	path to local directory, or compressed file (tar, zip, tar.gz tar.bz2, tgz2 or tbz)
subdir	subdirectory within url bundle that contains the R package.
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.

`upgrade` Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.

`force` Force installation, even if the remote state has not changed since the previous install.

`quiet` If TRUE, suppress output.

`build` If TRUE build the package before installing.

`build_opts` Options to pass to R CMD build, only used when build is TRUE.

`build_manual` If FALSE, don't build PDF manual ('–no-manual').

`build_vignettes` If FALSE, don't build package vignettes ('–no-build-vignettes').

`repos` A character vector giving repositories to use.

`type` Type of package to update.

`...` Other arguments passed on to utils::install.packages().

See Also

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_dev(), install_github(), install_gitlab(), install_git(), install_svn(), install_url(), install_version()

Examples

```
## Not run:
dir <- tempfile()
dir.create(dir)
pkg <- download.packages("testthat", dir, type = "source")
install_local(pkg[, 2])

## End(Not run)
```

install_svn *Install a package from a SVN repository*

Description

This function requires svn to be installed on your system in order to be used.

Usage

```
install_svn(
  url,
  subdir = NULL,
  args = character(0),
  revision = NULL,
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

url	Location of package. The url should point to a public or private repository.
subdir	A sub-directory within a svn repository that contains the package we are interested in installing.
args	A character vector providing extra options to pass on to svn.
revision	svn revision, if omitted updates to latest
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.

upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Force installation, even if the remote state has not changed since the previous install.
quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
repos	A character vector giving repositories to use.
type	Type of package to update.
...	Other arguments passed on to utils::install.packages().

Details

It is vectorised so you can install multiple packages with a single command.

See Also

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_dev(), install_github(), install_gitlab(), install_git(), install_local(), install_url(), install_version()

Examples

```
## Not run:
install_svn("https://github.com/hadley/stringr/trunk")
install_svn("https://github.com/hadley/httr/branches/oauth")

## End(Not run)
```

---

install_url *Install a package from a url*

Description

This function is vectorised so you can install multiple packages in a single command.

Usage

```
install_url(
  url,
  subdir = NULL,
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

url	location of package on internet. The url should point to a zip file, a tar file or a bzipped/gzipped tar file.
subdir	subdirectory within url bundle that contains the R package.
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.
upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Force installation, even if the remote state has not changed since the previous install.
quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.

build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
repos	A character vector giving repositories to use.
type	Type of package to update.
...	Other arguments passed on to utils::install.packages().

See Also

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_dev(), install_github(), install_gitlab(), install_git(), install_local(), install_svn(), install_version()

Examples

```
## Not run:
install_url("https://github.com/hadley/stringr/archive/HEAD.zip")

## End(Not run)
```

---

install_version *Install specific version of a package.*

Description

This function knows how to look in multiple CRAN-like package repositories, and in their archive directories, in order to find specific versions of the requested package.

Usage

```
install_version(
  package,
  version = NULL,
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = FALSE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = "source",
  ...
)
```

Arguments

package	Name of the package to install.
version	Version of the package to install. Can either be a string giving the exact version required, or a specification in the same format as the parenthesized expressions used in package dependencies. One of the following formats: • An exact version required, as a string, e.g. "0.1.13" • A comparison operator and a version, e.g. ">= 0.1.12" • Several criteria to satisfy, as a comma-separated string, e.g. ">= 1.12.0, < 1.14" • Several criteria to satisfy, as elements of a character vector, e.g. c(">= 1.12.0", "< 1.14")
dependencies	logical indicating whether to also install uninstalled packages which these packages depend on/link to/import/suggest (and so on recursively). Not used if repos = NULL. Can also be a character vector, a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). Only supported if lib is of length one (or missing), so it is unambiguous where to install the dependent packages. If this is not the case it is ignored, with a warning. The default, NA, means c("Depends", "Imports", "LinkingTo"). TRUE means to use c("Depends", "Imports", "LinkingTo", "Suggests") for pkgs and c("Depends", "Imports", "LinkingTo") for added dependencies: this installs all the packages needed to run pkgs, their examples, tests and vignettes (if the package author specified them correctly). In all of these, "LinkingTo" is omitted for binary packages.
upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Force installation, even if the remote state has not changed since the previous install.
quiet	logical: if true, reduce the amount of output. This is *not* passed to available.packages() in case that is called, on purpose.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
repos	character vector, the base URL(s) of the repositories to use, e.g., the URL of a CRAN mirror such as "https://cloud.r-project.org". For more details on supported URL schemes see url. Can be NULL to install from local files, directories or URLs: this will be inferred by extension from pkgs if of length one.

type	character, indicating the type of package to download and install. Will be "source" except on Windows and some macOS builds: see the section on 'Binary packages' for those.
...	Other arguments passed on to utils::install.packages().

Details

The repositories are searched in the order specified by the repos argument. This enables teams to maintain multiple in-house repositories with different policies - for instance, one repo for development snapshots and one for official releases. A common setup would be to first search the official release repo, then the dev snapshot repo, then a public CRAN mirror.

Older versions of packages on CRAN are usually only available in source form. If your requested package contains compiled code, you will need to have an R development environment installed. You can check if you do by running devtools::has_devel (you need the devtools package for this).

See Also

Other package installation: install_bioc(), install_bitbucket(), install_cran(), install_dev(), install_github(), install_gitlab(), install_git(), install_local(), install_svn(), install_url()

Examples

```
## Not run:
install_version("devtools", "1.11.0")
install_version("devtools", ">= 1.12.0, < 1.14")

## Specify search order (e.g. in ~/.Rprofile)
options(repos = c(
  prod = "http://mycompany.example.com/r-repo",
  dev = "http://mycompany.example.com/r-repo-dev",
  CRAN = "https://cran.revolutionanalytics.com"
))
install_version("mypackage", "1.15") # finds in 'prod'
install_version("mypackage", "1.16-39487") # finds in 'dev'

## End(Not run)
```

---

package_deps *Find all dependencies of a CRAN or dev package.*

Description

Find all the dependencies of a package and determine whether they are ahead or behind CRAN. A print() method identifies mismatches (if any) between local and CRAN versions of each dependent package; an update() method installs outdated or missing packages from CRAN.

Usage

```
package_deps(
  packages,
  dependencies = NA,
  repos = getOption("repos"),
  type = getOption("pkgType")
)

local_package_deps(pkgdir = ".", dependencies = NA)

dev_package_deps(
  pkgdir = ".",
  dependencies = NA,
  repos = getOption("repos"),
  type = getOption("pkgType")
)

## S3 method for class 'package_deps'
update(
  object,
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

packages	A character vector of package names.
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.

repos	A character vector giving repositories to use.
type	Type of package to update.
pkgdir	Path to a package directory, or to a package tarball.
object	A package_deps object.
upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Force installation, even if the remote state has not changed since the previous install.
quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
...	Additional arguments passed to install_packages.

Value

A data.frame with columns:

package	The dependent package's name,
installed	The currently installed version,
available	The version available on CRAN,
diff	An integer denoting whether the locally installed version of the package is newer (1), the same (0) or older (-1) t

Examples

```
## Not run:
package_deps("devtools")
# Use update to update any out-of-date dependencies
update(package_deps("devtools"))

## End(Not run)
```

parse-git-repo *Parse a remote git repo specification*

Description

A remote repo can be specified in two ways:

as a URL `parse_github_url()` handles HTTPS and SSH remote URLs and various GitHub browser URLs

via a shorthand `parse_repo_spec()` handles this concise form: `[username/]repo[/subdir][#pull|@ref|@*release]`

Usage

```
parse_repo_spec(repo)

parse_github_repo_spec(repo)

parse_github_url(repo)
```

Arguments

`repo`	Character scalar, the repo specification.

Value

List with members: `username`, `repo`, `subdir` `ref`, `pull`, `release`, some which will be empty.

Examples

```
parse_repo_spec("metacran/crandb")
parse_repo_spec("jimhester/covr#47")        ## pull request
parse_repo_spec("jeroen/curl@v0.9.3")       ## specific tag
parse_repo_spec("tidyverse/dplyr@*release") ## shorthand for latest release
parse_repo_spec("r-lib/remotes@550a3c7d3f9e1493a2ba") ## commit SHA
parse_repo_spec("igraph=igraph/rigraph") ## Different package name from repo name

parse_github_url("https://github.com/jeroen/curl.git")
parse_github_url("git@github.com:metacran/crandb.git")
parse_github_url("https://github.com/jimhester/covr")
parse_github_url("https://github.example.com/user/repo.git")
parse_github_url("git@github.example.com:user/repo.git")

parse_github_url("https://github.com/r-lib/remotes/pull/108")
parse_github_url("https://github.com/r-lib/remotes/tree/name-of-branch")
parse_github_url("https://github.com/r-lib/remotes/commit/1234567")
parse_github_url("https://github.com/r-lib/remotes/releases/latest")
parse_github_url("https://github.com/r-lib/remotes/releases/tag/1.0.0")
```

system_requirements *Query the system requirements for a package (and its dependencies)*

Description

Returns a character vector of commands to run that will install system requirements for the queried operating system.

Usage

```
system_requirements(
  os,
  os_release = NULL,
  path = ".",
  package = NULL,
  curl = Sys.which("curl")
)
```

Arguments

os, os_release	The operating system and operating system release version, see https://github.com/rstudio/r-system-requirements#operating-systems for the list of supported operating systems. If os_release is NULL, os must consist of the operating system and the version separated by a dash, e.g. "ubuntu-18.04".
path	The path to the dev package's root directory.
package	CRAN package name(s) to lookup system requirements for. If not NULL, this is used and path is ignored.
curl	The location of the curl binary on your system.

Value

A character vector of commands needed to install the system requirements for the package.

update_packages *Update packages that are missing or out-of-date.*

Description

Works similarly to utils::install.packages() but doesn't install packages that are already installed, and also upgrades out dated dependencies.

Usage

```
update_packages(
  packages = TRUE,
  dependencies = NA,
  upgrade = c("default", "ask", "always", "never"),
  force = FALSE,
  quiet = FALSE,
  build = TRUE,
  build_opts = c("--no-resave-data", "--no-manual", "--no-build-vignettes"),
  build_manual = FALSE,
  build_vignettes = FALSE,
  repos = getOption("repos"),
  type = getOption("pkgType"),
  ...
)
```

Arguments

packages	Character vector of packages to update.
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.
upgrade	Should package dependencies be upgraded? One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.
force	Deprecated, this argument has no effect.
quiet	If TRUE, suppress output.
build	If TRUE build the package before installing.
build_opts	Options to pass to R CMD build, only used when build is TRUE.
build_manual	If FALSE, don't build PDF manual ('–no-manual').
build_vignettes	If FALSE, don't build package vignettes ('–no-build-vignettes').
repos	A character vector giving repositories to use.

type	Type of package to update.
...	Other arguments passed on to utils::install.packages().

See Also

package_deps() to see which packages are out of date/ missing.

Examples

```
## Not run:
update_packages("ggplot2")
update_packages(c("plyr", "ggplot2"))

## End(Not run)
```

Index