Package 'pacs'

September 13, 2021

Title Supplementary Tools for R Packages Developers

Version 0.3.5

Maintainer Maciej Nasinski <nasinski.maciej@gmail.com>

Description Supplementary utils for CRAN maintainers and R packages developers.
Validating the library or packages.
Exploring a complexity of a specific package like evaluating sizes in bytes of all its dependencies.
Assessing the life duration of a specific package version.
Checking a CRAN package check page status for any errors and warnings.
Retrieving a DESCRIPTION or NAMESPACE file for any package version.
Comparing DESCRIPTION or NAMESPACE files between different package versions.
Getting a list of all releases for a specific package.
The Bioconductor is partly supported.

License GPL (>= 3)

URL https://github.com/Polkas/pacs

BugReports https://github.com/Polkas/pacs/issues

Encoding UTF-8

RoxygenNote 7.1.1

Depends R (>= 3.5.0)

Suggests remotes, withr, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports memoise, stringi, xml2

NeedsCompilation no

Author Maciej Nasinski [aut, cre]

Repository CRAN

Date/Publication 2021-09-13 19:10:02 UTC

R topics documented:

available_packages 2
biocran_repos 3

bio_releases . . . 3
checked_packages . . . 4
compareVersionsMax . . . 5
compareVersionsMin . . . 5
cran_flavors . . . 6
lib_validate . . . 6
pacs_base . . . 8
pac_checkpage . . . 9
pac_checkred . . . 9
pac_compare_namespace . . . 10
pac_compare_versions . . . 11
pac_deps . . . 13
pac_deps_timemachine . . . 14
pac_description . . . 15
pac_health . . . 16
pac_isin . . . 17
pac_islast . . . 18
pac_last . . . 19
pac_lifeduration . . . 19
pac_namespace . . . 21
pac_size . . . 22
pac_timemachine . . . 22
pac_true_size . . . 23
pac_validate . . . 24

Index **27**

`available_packages` *List Available Packages at CRAN-like Repositories*

Description

available_packages returns a matrix of details corresponding to packages currently available at one or more repositories. The current list of packages is downloaded over the internet (or copied from a local mirror).

Usage

```
available_packages(repos = biocran_repos())
```

Arguments

`repos`	character vector, the base URL(s) of the repositories to use. Default `pacs::biocran_repos()`

biocran_repos *CRAN and Bioconductor repositories*

Description

CRAN and Bioconductor repositories. The newest Bioconductor release for the specific R version is assumed.

Usage

```
biocran_repos(version = NULL)
```

Arguments

version	character the Bioconductor release. By default the newest Bioconductor release for the specific R version is assumed, if not available only CRAN repository is returned. Available Bioconductor versions for your R version could be checked with `pacs::bio_releases()`. Default NULL

Value

named character vector of repositories.

Examples

```
## Not run:
biocran_repos()

## End(Not run)
```

bio_releases *Retrieving all Bioconductor releases*

Description

Retrieving all Bioconductor releases. The data is downloaded from https://www.bioconductor.org/about/release-announcements/.

Usage

```
bio_releases()
```

Value

data.frame with the same structure as the html table on https://www.bioconductor.org/about/release-announcements/.

Note

Results are cached for 1 hour with `memoise` package.

Examples

```
## Not run:
bio_releases()

## End(Not run)
```

`checked_packages` *Retrieving all R CRAN packages check pages statuses.*

Description

Retrieving all R CRAN packages check pages statuses. The data is downloaded from https://cran.r-project.org/web/checks/check_summary_by_package.html.

Usage

```
checked_packages()
```

Value

data.frame with the same structure as the html table on https://cran.r-project.org/web/checks/check_summary_by_package.ht

Note

Results are cached for 1 hour with `memoise` package. Some packages could be duplicated as not all tests are performed for a new version so two versions still coexists. Checks with asterisks (*) indicate that checking was not fully performed, this is a case for less than 1% of all packages.

Examples

```
## Not run:
checked_packages()

## End(Not run)
```

compareVersionsMax *Maximum version across the vector*

Description

Reduce function over the `utils::compareVersion`

Usage

```
compareVersionsMax(vec, na.rm = TRUE)
```

Arguments

`vec`	character vector
`na.rm`	logical if to remove NA values.

Value

character maximum version

Examples

```
compareVersionsMax(c("1.1.1", "0.2.0"))
```

compareVersionsMin *Minimum version across the vector*

Description

Reduce function over the `utils::compareVersion`

Usage

```
compareVersionsMin(vec, na.rm = TRUE)
```

Arguments

`vec`	character vector
`na.rm`	logical if to remove NA values.

Value

character minimal version

Examples

```
compareVersionsMin(c("1.1.1", "0.2.0"))
```

cran_flavors *Retrieving all R CRAN servers flavors*

Description

Retrieving all R CRAN servers flavors. The data is downloaded from https://cran.r-project.org/web/checks/check_flavors.htm

Usage

```
cran_flavors()
```

Value

data.frame with the same structure as the html table on https://cran.r-project.org/web/checks/check_flavors.html.

Note

Results are cached for 1 hour with `memoise` package.

Examples

```
## Not run:
cran_flavors()

## End(Not run)
```

lib_validate *Validate the local library*

Description

Checking if installed packages have correct versions taking into account all DESCRIPTION files requirements. Moreover identifying which packages are newest releases. Optionally we could add life duration and CRAN check page status for each package.

Usage

```
lib_validate(
  lib.loc = NULL,
  fields = c("Depends", "Imports", "LinkingTo"),
  lifeduration = FALSE,
  checkred = list(scope = character(0), flavors = NULL),
  repos = biocran_repos()
)
```

Arguments

lib.loc	character. Default: NULL
fields	character vector with possible values c("Depends","Imports","LinkingTo","Suggests"). Default: c("Depends","Imports","LinkingTo")
lifeduration	logical if to add life duration column, might take some time. Default: FALSE
checkred	list with two named fields, scope and flavor. scope of R CRAN check pages statuses to consider, any of c("ERROR","FAIL","WARN","NOTE"). flavor is a vector of CRAN machines to consider, which might be retrieved with pacs::cran_flavors()$Flavor. By default an empty scope field deactivated assessment for checkred column, and NULL flavor will results in checking all machines. Default list(scope = character(0),flavor = NULL)
repos	character vector base URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos()

Value

data.frame with 6/7/8 columns.

Package character a package name.

Version.expected.min character expected by DESCRIPTION files minimal version. "" means not specified so the newest version.

Version.have character installed package version.

version_status numeric -1/0/1 which comes from utils::compareVersion function. 0 means that we have the same version as required by DESCRIPTION files. -1 means we have too low version installed, this is an error. 1 means we have higher version.

newest logical if the installed version is the newest one. For Bioconductor if is the newest one per R version.

cran logical if the package is on CRAN, version is not taken into account here.

checkred (Optional) logical if the NEWEST package contains any specified statuses on CRAN check page. pacs::checked_packages is used to quickly retrieve all statuses at once.

lifeduration (Optional) integer number of days a package was released.

Note

Version.expected.min column not count packages which are not a dependency for any package, so could not be find in DESCRIPTION files. When turn on the lifeduration options, calculations might be time consuming. Results are cached for 1 hour with memoise package. BioConductor packages are tested only in available scope, checkred is not assessed for them.

Examples

```
## Not run:
lib_validate()
lib_validate(checkred = list(scope = c("ERROR", "FAIL", "WARN")))
lib_validate(checkred = list(
  scope = c("ERROR", "FAIL"),
```

```
  flavors = cran_flavors()$Flavor[1:2]
))
# activate lifeduration argument, could be time consuming for bigger libraries.
lib_validate(
  lifeduration = TRUE,
  checkred = list(scope = c("ERROR", "FAIL"))
)
# only R CRAN repository
lib_validate(repos = "https://cran.rstudio.com/")

## End(Not run)
```

---

pacs_base *List of base R packages*

Description

using installed.packages and priority equal "base" to retrieve base packages.

Usage

```
pacs_base(startup = FALSE)
```

Arguments

startup	logical include only startup packages. Default: FALSE

Value

character vector

Examples

```
## Not run:
pacs_base()
pacs_base(startup = TRUE)

## End(Not run)
```

pac_checkpage *Retrieving the R CRAN package check page*

Description

Retrieving the R CRAN package check page.

Usage

```
pac_checkpage(pac)
```

Arguments

pac	character a package name.

Value

data.frame.

Note

Results are cached for 1 hour with `memoise` package. If you need to check many packages at once then is recommended usage of `pacs::checked_packages`. Please as a courtesy to the R CRAN, don't overload their server by constantly using this function.

Examples

```
## Not run:
pac_checkpage("dplyr")

## End(Not run)
```

pac_checkred *Checking the R CRAN package check page status*

Description

using package R CRAN check page to validate if there are ANY errors and/or fails and/or warnings and/or notes.

Usage

```
pac_checkred(pac, scope = c("ERROR", "FAIL"), flavors = NULL)
```

Arguments

pac	character a package name.
scope	character vector scope of the check, accepted values c("ERROR", "FAIL", "WARN", "NOTE"). Default c("ERROR", "FAIL")
flavors	character vector of CRAN machines to consider, which might be retrieved with `pacs::cran_flavors()$Flavor`. By default all CRAN machines are considered, NULL value. Default NULL

Value

logical if the package fail under specified criteria.

Note

Results are cached for 1 hour with `memoise` package. If you need to check many packages at once then is recommended usage of `pacs::checked_packages`. The used repository https://cran.rstudio.com/. Please as a courtesy to the R CRAN, don't overload their server by constantly using this function.

Examples

```
## Not run:
pac_checkred("dplyr")
pac_checkred("dplyr", scope = c("ERROR"))
pac_checkred("dplyr",
  scope = c("ERROR", "FAIL", "WARN"),
  flavors = c(
    "r-devel-linux-x86_64-debian-clang",
    "r-devel-linux-x86_64-debian-gcc"
  )
)

## End(Not run)
```

---

pac_compare_namespace *Compare NAMESPACE exports between specific CRAN packages versions*

Description

using the remote github CRAN mirror to compare NAMESPACE exports between specific packages versions.

Usage

```
pac_compare_namespace(
  pac,
  old = NULL,
  new = NULL,
  lib.loc = NULL,
  repos = "https://cran.rstudio.com/"
)
```

Arguments

pac	character a package name.
old	character an old version of package.
new	character a new version of package.
lib.loc	character. Default: NULL
repos	character the base URL of the CRAN repository to use. Used only for the validation. Default https://cran.rstudio.com/

Value

list with c("imports","exports","exportPatterns","importClasses","importMethods","exportClasses","expor
slots, and added and removed ones for each of them.

Examples

```
## Not run:
pac_compare_namespace("shiny", "1.0.0", "1.6.0")
pac_compare_namespace("shiny", "1.0.0", "1.6.0")$exports
# local version to newest one
pac_compare_namespace("shiny")

## End(Not run)
```

---

pac_compare_versions *Compare DESCRIPTION files dependencies between specific CRAN packages versions*

Description

using the remote github CRAN mirror to compare DESCRIPTION files dependencies between specific packages versions.

Usage

```
pac_compare_versions(
  pac,
  old = NULL,
  new = NULL,
  fields = c("Imports", "Depends", "LinkingTo"),
  lib.loc = NULL,
  repos = "https://cran.rstudio.com/"
)
```

Arguments

pac	character a package name.
old	character an old version of package, default local version. Default: NULL
new	character a new version of package, default newest version. Default: NULL
fields	character a vector with possible values `c("Depends","Imports","LinkingTo","Suggests")`. Default: `c("Depends","Imports","LinkingTo")`
lib.loc	character. Default: NULL
repos	character the base URL of the CRAN repository to use. Default "https://cran.rstudio.org"

Value

data.frame with 4 columns.

Package character package names.

Version.OLD character versions of dependencies required by an old package version.

Version.NEW character versions of dependencies required by a new package version.

version_status numeric -1/0/1 which comes from `utils::compareVersion` function. 0 means that we have the same version as required by DESCRIPTION files. -1 means we have too low version installed, this is an error. 1 means we have higher version.

Examples

```
## Not run:
pac_compare_versions("memoise", "0.2.1", "2.0.0")
pac_compare_versions("memoise", "0.2.1")
# local version to newest one
pac_compare_versions("memoise")

## End(Not run)
```

pac_deps *package dependencies*

Description

Package dependencies from DESCRIPTION files with installed or expected versions or newest released.

Usage

```
pac_deps(
  pac,
  fields = c("Depends", "Imports", "LinkingTo"),
  lib.loc = NULL,
  base = FALSE,
  local = TRUE,
  description_v = FALSE,
  attr = TRUE,
  recursive = TRUE,
  repos = "https://cran.rstudio.com/"
)
```

Arguments

pac	character a package name.
fields	character vector with possible values c("Depends","Imports","LinkingTo","Suggests"). Default: c("Depends","Imports","LinkingTo")
lib.loc	character vector. Is omitted for non NULL version. Default: NULL
base	logical if to add base packages too. Default: FALSE
local	logical if to use newest CRAN packages, where by default local ones are used. Default: TRUE
description_v	if the dependencies version should be taken from description files, minimal required. Default: FALSE
attr	logical specify if a package and its version should be added as a attribute of data.frame or for FALSE as a additional record. Default: TRUE
recursive	logical if to assess the dependencies recursively. Default: TRUE
repos	character the base URL of the CRAN repository to use. Used only for the validation. Default https://cran.rstudio.com/

Value

data.frame with packages and their versions. Versions are taken from installed.packages or newest released.

Note

When function is invoked in the loop afterwards binded results could be aggregated like, `stats::aggregate(results[,c("V = FALSE],list(Package = results$Package),pacs::compareVersionsMax)`.

Examples

```
## Not run:
pacs::pac_deps("stats", base = TRUE)$Package
pacs::pac_deps("memoise")$Package
pacs::pac_deps("memoise", description_v = FALSE)
# raw dependencies from DESCRIPTION file
pacs::pac_deps("memoise", description_v = TRUE, recursive = FALSE)
# raw dependencies from DESCRIPTION file - last release
pacs::pac_deps("memoise", description_v = TRUE, local = FALSE, recursive = FALSE)

## End(Not run)
```

`pac_deps_timemachine` *R CRAN package dependencies for a certain version or time point*

Description

Package dependencies from DESCRIPTION files retrieved recursively for certain version or time point.

Usage

```
pac_deps_timemachine(
  pac,
  version = NULL,
  at = NULL,
  fields = c("Depends", "Imports", "LinkingTo"),
  recursive = TRUE
)
```

Arguments

`pac`	character a package name.
`version`	character version of package. Default: NULL
`at`	Date old version of package. Default: NULL
`fields`	character vector with possible values `c("Depends","Imports","LinkingTo","Suggests")`. Default: `c("Depends","Imports","LinkingTo")`
`recursive`	logical if to assess the dependencies recursively. Default: TRUE

Value

named vector package dependencies and their versions at the release date of main package plus one day.

Note

Longer lived version is taken if 2 is available at the same date (switch time).

Examples

```
## Not run:
pacs::pac_deps_timemachine("memoise", "0.2.1")
pacs::pac_deps_timemachine("memoise", at = as.Date("2019-01-01"))

## End(Not run)
```

---

pac_description *package DESCRIPTION file*

Description

CRAN package DESCRIPTION file taken locally or remotely from GITHUB CRAN mirror or CRAN website.

Usage

```
pac_description(
  pac,
  version = NULL,
  at = NULL,
  local = FALSE,
  lib.loc = NULL,
  repos = "https://cran.rstudio.com/"
)
```

Arguments

pac	character a package name.
version	character package version, By default the newest version in taken if failed tried to give local one if installed. Default: NULL
at	Date. Default: NULL
local	logical if to use local library. Default: FALSE
lib.loc	character used optionally when local is equal TRUE. Default: NULL
repos	character the base URL of the CRAN repository to use. Used only for the validation. Default https://cran.rstudio.com/

Value

list with names proper for DESCRIPTION file fields.

Note

Results are cached for 1 hour with `memoise` package.

Examples

```
## Not run:
pac_description("dplyr", version = "0.8.0")
pac_description("dplyr", at = as.Date("2019-02-01"))

## End(Not run)
```

`pac_health` *CRAN package health state at a specific Date or for a specific version*

Description

using CRAN website to get a package version/versions used at a specific Date interval. A healthy package was published for more than x days, where default is 14 days. CRAN team gives around one/two week to resolved a package which gave errors under the check page.

Usage

```
pac_health(
  pac,
  version = NULL,
  at = NULL,
  limit = 14,
  scope = c("ERROR", "FAIL"),
  flavors = NULL,
  lib.loc = NULL,
  repos = "https://cran.rstudio.com/"
)
```

Arguments

`pac`	character a package name.
`version`	character package version, By default the newest version is taken. Default: NULL
`at`	Date old version of package. Default: NULL
`limit`	numeric at least days to treat as healthy. Default: 14
`scope`	character vector scope of R CRAN check pages statuses to consider, any of `c("ERROR","FAIL","WARN","NOTE")`. Default `c("ERROR","FAIL")`
`flavors`	character vector of CRAN machines to consider, which might be retrieved with `pacs::cran_flavors()$Flavor`. By default all CRAN machines are considered, NULL value. Default NULL
`lib.loc`	character vector. Is omitted for non NULL version. Default: NULL
`repos`	character the base CRAN URL of the repository to use. Default "https://cran.rstudio.org"

Value

logical if package is healthy.

Note

Function will scrap two/tree CRAN URLS. Works only with CRAN packages. The newest release are checked for warnings/errors on R CRAN check page. Please as a courtesy to the R CRAN, don't overload their server by constantly using this function. Results are cached for 1 hour with `memoise` package, memory cache.

Examples

```
## Not run:
pac_health("memoise")
pac_health("dplyr", version = "0.8.0", limit = 14)
pac_health("dplyr", limit = 14, scope = c("ERROR", "FAIL"))

## End(Not run)
```

---

pac_isin *Checking if a package is in repositories*

Description

using `utils::available.packages` to check if package is in repositories.

Usage

```
pac_isin(pac, repos = biocran_repos())
```

Arguments

`pac`	character a package name.
`repos`	character vector base URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default `pacs::biocran_repos()`

Value

logical if a package is inside repositories.

Note

Results are cached for 1 hour with `memoise` package.

Examples

```
## Not run:
pac_isin("dplyr")
pac_isin("dplyr", repos = "https://cran.rstudio.com/")
pac_isin("dplyr", repos = biocran_repos()[grep("Bio", names(biocran_repos()))])

## End(Not run)
```

`pac_islast` *Checking if a package version is the most recent one*

Description

checking if a package version is the most recent one, by default the installed version is compared.

Usage

```
pac_islast(pac, version = NULL, lib.loc = NULL, repos = biocran_repos())
```

Arguments

`pac`	character a package name.
`version`	character package version, by default the installed version is taken. Default: NULL
`lib.loc`	character vector. Is omitted for non NULL version. Default: NULL
`repos`	character vector base URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R Version. Default `pacs::biocran_repos()`

Value

logical if a package is inside repositories.

Note

Results are cached for 1 hour with `memoise` package. For Bioconductor if package is the newest one per R version.

Examples

```
## Not run:
pac_islast("memoise")
pac_islast("dplyr", version = "1.0.0")
pac_islast("S4Vectors")
pac_islast("S4Vectors", version = pac_last("S4Vectors"))

## End(Not run)
```

pac_last *Getting the most recent package version*

Description

using `utils::available.packages` to get the newest package version.

Usage

```
pac_last(pac, repos = biocran_repos())
```

Arguments

`pac`	character a package name.
`repos`	character vector base URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R Version. Default `pacs::biocran_repos()`

Value

character most recent package version.

Note

Results are cached for 1 hour with `memoise` package. For Bioconductor the newest one per R version.

Examples

```
## Not run:
pac_last("dplyr")
pac_last("S4Vectors")

## End(Not run)
```

pac_lifeduration *Package version life duration at specific Date or for a specific version*

Description

using CRAN website to get a package life duration for certain version or at a specific Date.

Usage

```
pac_lifeduration(
  pac,
  version = NULL,
  at = NULL,
  lib.loc = NULL,
  repos = biocran_repos()
)
```

Arguments

pac	character a package name.
version	character package version, By default the newest version is taken. Default: NULL
at	Date old version of package. Default: NULL
lib.loc	character vector. Is omitted for non NULL version. Default: NULL
repos	character vector base URLs of the repositories to use. By default checking CRAN and newest Bioconductor. Default `pacs::biocran_repos()`

Value

`difftime`, number of days package version was the newest one.

Note

Function will scrap two github CRAN mirror and CRAN URL. Works mainly with CRAN packages. Please as a courtesy to the R CRAN, don't overload their server by constantly using this function. Results are cached for 1 hour with `memoise` package, memory cache.

Examples

```
## Not run:
pac_lifeduration("memoise")
pac_lifeduration("dplyr", version = "0.8.0")
pac_lifeduration("dplyr", at = as.Date("2019-02-14"))
# For Bioconductor packages will work only for the newest per R version and installed packages.
pac_lifeduration("S4Vectors")

## End(Not run)
```

pac_namespace *package NAMESPACE file*

Description

CRAN package NAMESPACE file taken locally or remotely from GITHUB CRAN mirror or CRAN website.

Usage

```
pac_namespace(
  pac,
  version = NULL,
  at = NULL,
  local = FALSE,
  lib.loc = NULL,
  repos = "https://cran.rstudio.com/"
)
```

Arguments

pac	character a package name.
version	character package version, By default the newest version in taken if failed tried to give local one if installed. Default: NULL
at	Date. Default: NULL
local	logical if to use local library. Default: FALSE
lib.loc	character used optionally when local is equal TRUE. Default: NULL
repos	character the base URL of the CRAN repository to use. Used only for the validation. Default https://cran.rstudio.com/

Value

list with names proper for NAMESPACE file, the same as format as returned by `base::parseNamespaceFile`.

Note

Results are cached for 1 hour with `memoise` package. This function is mainly built under source code from `base::parseNamespaceFile`.

Examples

```
## Not run:
pac_namespace("dplyr", version = "0.8.0")
pac_namespace("dplyr", at = as.Date("2019-02-01"))
pac_namespace("memoise", local = TRUE)

## End(Not run)
```

pac_size *Size of the package*

Description

size of package.

Usage

```
pac_size(pac, lib.loc = NULL)
```

Arguments

pac	character a package name.
lib.loc	character vector. Default: NULL

Value

numeric size in bytes, to get MB ten divide by `10**6`.

Examples

```
## Not run:
cat(pacs::pac_size("stats") / 10**6, "MB")

## End(Not run)
```

pac_timemachine *Package metadata for all releases*

Description

Using CRAN website to get a package metadata used at a specific Date or a Date interval or for specific version.

Usage

```
pac_timemachine(pac, at = NULL, from = NULL, to = NULL, version = NULL)
```

Arguments

pac	character a package name.
at	Date old version of package. Default: NULL
from	Date new version of package. Default: NULL
to	Date CRAN URL. Default: NULL
version	character version of package. Default: NULL

Value

data.frame with 7 columns

Package character package name.

Version character package version.

Released character release Date

Archived character archived Date.

LifeDuration `difftime` number of days the version was the newest one.

URL character the suffix of the base URL to tar.gz file.

Size character size of the `tar.gz` file.

Note

Function will scrap two CRAN URLS. Works only with CRAN packages. Please as a courtesy to the R CRAN, don't overload their servers by constantly using this function. The base part of URL in the result is https://cran.r-project.org/src/contrib/. Results are cached for 1 hour with `memoise` package.

Examples

```
## Not run:
pac_timemachine("dplyr", at = as.Date("2017-02-02"))
pac_timemachine("dplyr", from = as.Date("2017-02-02"), to = as.Date("2018-04-02"))
pac_timemachine("dplyr", at = Sys.Date())
pac_timemachine("tidyr", from = as.Date("2020-06-01"), to = Sys.Date())

## End(Not run)
```

`pac_true_size` *True size of the package*

Description

True size of the package as it takes into account its dependencies.

Usage

```
pac_true_size(
  pac,
  fields = c("Depends", "Imports", "LinkingTo"),
  lib.loc = NULL,
  exclude_joint = 0L
)
```

Arguments

pac	character a package name.
fields	character vector, Default: c("Depends","Imports","LinkingTo")
lib.loc	character vector, Default: NULL
exclude_joint	integer exclude packages which are dependencies of at least N other packages, not count main package dependencies. Default: 0

Value

numeric size in bytes, to get MB then divide by 10**6.

Note

R base packages are not counted.

Examples

```
## Not run:
# size in MB, with all its dependencies
pacs::pac_true_size("memoise") / 10**6

## End(Not run)
```

---

pac_validate *Validate a specific local package*

Description

Checking if installed package dependencies have correct versions taking into account their DESCRIPTION files requirements. Moreover identifying which packages are newest releases. Optionally we could add life duration and CRAN check page status for each dependency.

Usage

```
pac_validate(
  pac,
  lib.loc = NULL,
  fields = c("Depends", "Imports", "LinkingTo"),
  lifeduration = FALSE,
  checkred = list(scope = character(0), flavors = NULL),
  repos = biocran_repos()
)
```

Arguments

pac	character a package name.
lib.loc	character. Default: NULL
fields	character vector with possible values c("Depends","Imports","LinkingTo","Suggests"). Default: c("Depends","Imports","LinkingTo")
lifeduration	logical if to add life duration column, might take some time. Default: FALSE
checkred	list with two named fields, scope and flavor. scope of R CRAN check pages statuses to consider, any of c("ERROR","FAIL","WARN","NOTE"). flavor vector of machines to consider, which might be retrieved with pacs::cran_flavors()$Flavor. By default an empty scope field deactivated assessment for checkred column, and NULL flavor will results in checking all machines. Default list(scope = character(0),flavor = NULL)
repos	character vector base URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos()

Value

data.frame with 5/6/7 columns.

Package character a package name.

Version.expected.min character expected by DESCRIPTION files minimal version. "" means not specified so the newest version.

Version.have character installed package version.

version_status numeric -1/0/1 which comes from utils::compareVersion function. 0 means that we have the same version as required by DESCRIPTION files. -1 means we have too low version installed, this is an error. 1 means we have higher version.

direct logical if the package is in the first depencency layer, direct depencencies from DESCRIPTION file.

newest logical if the installed version is the newest one.

cran logical if the package is on CRAN, version is not taken into account here.

checkred (Optional) logical if the NEWEST package contains any specified statuses on CRAN check page.

lifeduration (Optional) integer number of days a package was released.

Note

Version.expected.min column not count packages which are not a dependency for any package, so could not be find in DESCRIPTION files. When turn on the lifeduration option, calculations might be time consuming. Please as a courtesy to the R CRAN, don't overload their server by constantly using this function with lifeduration or checkred turned on. Results are cached with memoise package, memory cache.

Examples

```
## Not run:
pac_validate("memoise")
pac_validate("memoise",
  lifeduration = TRUE,
  checkred = list(scope = c("ERROR", "FAIL"), flavors = NULL)
)

## End(Not run)
```

Index

available_packages, 2

bio_releases, 3
biocran_repos, 3

checked_packages, 4
compareVersionsMax, 5
compareVersionsMin, 5
cran_flavors, 6

lib_validate, 6

pac_checkpage, 9
pac_checkred, 9
pac_compare_namespace, 10
pac_compare_versions, 11
pac_deps, 13
pac_deps_timemachine, 14
pac_description, 15
pac_health, 16
pac_isin, 17
pac_islast, 18
pac_last, 19
pac_lifeduration, 19
pac_namespace, 21
pac_size, 22
pac_timemachine, 22
pac_true_size, 23
pac_validate, 24
pacs_base, 8