Package ‘pacs’

May 11, 2022

Title Supplementary Tools for R Packages Developers
Version 0.4.5
Maintainer Maciej Nasinski <nasinski.maciej@gmail.com>
Description Supplementary utils for CRAN maintainers and R packages developers.
Validating the library, packages and lock files.
Exploring a complexity of a specific package like evaluating its size in bytes with all dependencies.
The shiny app complexity could be explored too.
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)
BugReports https://github.com/Polkas/pacs/issues
Encoding UTF-8
RoxygenNote 7.1.2
Depends R (>= 3.5.0)
Imports curl, memoise, jsonlite, xml2, stringi
Suggests nomnoml, remotes, renv, withr, pkgsearch, posterdown,
mockery, testthat (>= 3.0.0), knitr, rmarkdown
Config/testthat/edition 3
VignetteBuilder knitr
NeedsCompilation no
Author Maciej Nasinski [aut, cre]
Repository CRAN
Date/Publication 2022-05-10 22:50:02 UTC
R topics documented:

app_deps ................................................................. 2
app_size ................................................................. 4
biocran_repos ............................................................ 5
bio_releases ............................................................... 5
checked_packages ......................................................... 6
compareVersionsMax ......................................................... 7
compareVersionsMin ......................................................... 7
cran_flavors .............................................................. 8
dir_size ................................................................. 8
lib_validate ............................................................... 9
lock_validate ............................................................. 11
pacs_base .............................................................. 13
pacs_lifeduration .......................................................... 13
pac_checkpage .......................................................... 14
pac_checkred ............................................................ 15
pac_compare_namespace .................................................... 16
pac_compare_versions ..................................................... 17
pac_deps ................................................................. 18
pac_deps_dev ............................................................. 20
pac_deps_timemachine ...................................................... 21
pac_deps_user ............................................................. 22
pac_description .......................................................... 23
pac_health ............................................................... 24
pac_isin ................................................................. 25
pac_islast ............................................................... 26
pac_last ................................................................. 27
pac_lifeduration .......................................................... 27
pac_namespace ........................................................... 29
pac_size ................................................................. 30
pac_timemachine ........................................................... 30
pac_true_size ............................................................. 32
pac_validate ............................................................. 33

Index ............................................................................. 35

---

app_deps

The shiny app dependencies

Description

the shiny app dependencies packages are checked recursively. The c("Depends", "Imports", "LinkingTo") DESCRIPTION files fields are checked recursively. The required dependencies have to be installed in the local repository. The default arguments setup is recommended.
Usage

```r
app_deps(
  path = ".",
  fields = c("Depends", "Imports", "LinkingTo"),
  lib.loc = .libPaths(),
  local = TRUE,
  base = FALSE,
  description_v = FALSE,
  recursive = TRUE,
  repos = biocran_repos()
)
```

Arguments

- `path` path to the shiny app. Default: "."
- `fields` a character vector listing the types of dependencies, a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: c("Depends", "Imports", "LinkingTo")
- `lib.loc` character vector, used optionally when local is equal TRUE. Default: .libPaths()
- `local` logical if to use local repository or newest CRAN packages, where by default local packages are used. Default: TRUE
- `base` logical if to add base packages too. If TRUE then pacs::pacs_base() are taken into account. Default: FALSE
- `description_v` if the dependencies version should be taken from description files, minimal required. By default installed versions are taken. Default: FALSE
- `recursive` logical if to assess the dependencies recursively. Default: TRUE
- `repos` character vector URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos()

Value

character vector with dependency packages or data.frame when checking recursively.

Note

renv package has to be installed.

Examples

```r
## Not run:
library(renv)
# Please update the path to the custom shiny app
app_path <- system.file("examples/04_mpg", package = "shiny")
pacs::app_deps(app_path)
pacs::app_deps(app_path, recursive = FALSE)
```
app_size

Size of the shiny app

Description

The size of shiny app is a sum of dependencies packages and the app directory. The app dependencies packages are checked recursively, and only in local repository. The default arguments setup is recommended.

Usage

```r
app_size(
  path = ".", 
  fields = c("Depends", "Imports", "LinkingTo"), 
  lib.loc = .libPaths(), 
  recursive = TRUE
)
```

Arguments

- `path` path to the shiny app. Default: "."
- `fields` a character vector listing the types of dependencies, a subset of `c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")`. Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: `c("Depends", "Imports", "LinkingTo")`
- `lib.loc` character vector, used optionally when local is equal TRUE. Default: `.libPaths()`
- `recursive` logical if to assess the dependencies recursively. Default: TRUE

Value

numeric size in bytes, to get MB ten divide by $10^6$.

Note

`renv` package has to be installed. base packages (`pacs::pacs_base()`) are not taken into account.

Examples

```r
## Not run:
library(renv)
# Please update the path to the shiny app
cat(pacs::app_size(system.file("examples/04_mpg", package = "shiny")) / 10**6, "MB")
```

## End(Not run)
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.

**Note**

The Internet connection is needed to get Bioconductor repositories.

**Examples**

```r
## 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 30 minutes with memoise package.

Examples

```r
## Not run:
pacs::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.html.

Note

Results are cached for 30 minutes 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

```r
## Not run:
pacs::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.html.

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 30 minutes with memoise package.

Examples

```r
## Not run:
pacs::cran_flavors()
## End(Not run)
```

dir_size  

Size of the package

Description

size of package.

Usage

dir_size(path = ".", recursive = TRUE)

Arguments

- `path` path to the directory. Default: "."
- `recursive` logical if to assess the dependencies recursively. Default: TRUE

Value

numeric size in bytes, to get MB ten divide by \(10^6\).
**Examples**

```r
## Not run:
cat(pacs::dir_size(system.file(package = "stats")) / 10**6, "MB")
## End(Not run)
```

**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**

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

**Arguments**

- `lib.loc` character vector. Default: `.libPaths()`
- `fields` a character vector listing the types of dependencies, a subset of `c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")`. Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: `c("Depends", "Imports", "LinkingTo")`
- `lifeduration` logical if to assess life duration for each package in the library. MEATCRAN CRANDB is used for libraries with less than 500 packages. The direct web page download from CRAN or local evaluation for newest packages otherwise. 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`) `built` logical if to add an R version under which each package was installed. Useful mainly for a local usage. Packages installed with a previous version of R could not work correctly with the new version of R. Default: FALSE
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 4/6/8/9/10 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.

**built** character package was built under this R version

**built_status** integer if the package was built under the current R version, then 1 (good) and for older R versions 0 (possibly bad). A package built under older R version or mix of packages built under different versions could bring possible failures.

**newest** logical (Internet needed) if the installed version is the newest one. For Bioconductor if is the newest one per R version.

**cran** logical (Internet needed) if the package is on CRAN, version is not taken into account here.

**checkred** (Optional) (Internet needed) 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) (Internet needed) 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 for libraries bigger than 500 packages. Results are cached for 30 minutes with memoise package. BioConductor packages are tested only in available scope, checkred is not assessed for them. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/.

Examples

```r
# Not run:
pacs::lib_validate()
pacs::lib_validate(checkred = list(scope = c("ERROR", "FAIL", "WARN")))
pacs::lib_validate(checkred = list(
  scope = c("ERROR", "FAIL"),
  flavors = cran_flavors()$Flavor[1:2]
))
# activate lifeduration argument, could be time consuming for bigger libraries.
pacs::lib_validate(
```
 validating

lifeduration = TRUE,  
checkred = list(scope = c("ERROR", "FAIL"))
)
# only R CRAN repository
pacs::lib_validate(repos = "https://cran.rstudio.com/")
## End(Not run)

lock_validate  Validate a specific renv lock file

Description

This function will be especially useful when renv lock file is built manually. Checking if packages in the lock file have correct versions taking into account their DESCRIPTION files requirements (c("Depends", "Imports", "LinkingTo")). Moreover identifying which packages are newest releases. Optionally we could add life duration and CRAN check page status for each dependency.

Usage

lock_validate(  
  path,  
  lifeduration = FALSE,  
  checkred = list(scope = character(0), flavors = NULL),  
  lib.loc = .libPaths(),  
  repos = biocran_repos()
)

Arguments

path  character a path to the renv lock file, or url.

lifeduration  logical if to assess life duration for each package in the library. MEATCRAN CRANDB is used for less than 500 packages. The direct web page download from CRAN or local evaluation for newest packages otherwise. 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)

lib.loc  character vector. Default: .libPaths()

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 2/6/7/8 columns.

Package character a package name.

Version.expected.min (conditional) (Internet needed) character expected by DESCRIPTION files minimal version. "" means not specified so the newest version.

Version.expected character package version in the renv lock file.

version_status (conditional) 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 (Internet needed) if the installed version is the newest one.

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

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

lifeduration (Optional) (Internet needed) 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. Version.expected.min and version_status are assessed only if there are less than 500 packages in the lock file. When turn on the lifeduration option, calculations might be time consuming when there is more than 500 packages. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/.

Examples

```r
## Not run:
# path or url
url <- "https://raw.githubusercontent.com/Polkas/pacs/master/tests/testthat/files/renv_test.lock"
pacs::lock_validate(url)

pacs::lock_validate(
  url,
  checkred = list(scope = c("ERROR", "FAIL"), flavors = NULL)
)

pacs::lock_validate(
  url,
  lifeduration = TRUE,
  checkred = list(scope = c("ERROR", "FAIL"), flavors = NULL)
)

## End(Not run)
```
**pacs_base**  

List of base R packages

Description

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

Usage

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

Arguments

- **startup**  
  logical include only startup packages. Default: FALSE

Value

character vector

Examples

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

**pacs_lifeduration**  

Packages life duration for a specific version

Description

packages life duration for certain versions.

Usage

```r
pacs_lifeduration(
pacs,
versions,
source = c("crandb", "loop_crandb", "loop_cran"),
lib.loc = .libPaths(),
repos = biocran_repos()
)
```
Arguments

- **pacs**: character vector packages names.
- **versions**: character vector packages versions.
- **source**: character one of c("crandb", "loop_cranb", "loop_cran"). The "crandb" works if less than getOption("pacs.crandb_limit") (currently 500) packages are looked for. Default: "crandb"
- **lib.loc**: character vector. Is omitted for non NULL version. Default: .libPaths()
- **repos**: character vector repositories. Default https://cran.rstudio.org

Value

data.frame with two columns package name and life duration.

Note

Results are cached for 30 minutes with memoise package. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/. For source = "loop_cran" the 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.

Examples

```r
## Not run:
pacs::pacs_lifeduration(c("dplyr", "tidyr"), c("1.0.0", "1.2.0"))
pacs::pacs_lifeduration(c("dplyr", "tidyr"), c("1.0.0", "1.2.0"), source = "loop_cran")
# last versions
pacs::pac_lifeduration(c("dplyr", "tidyr"), sapply(c("dplyr", "tidyr"), pacs::pac_last))
## 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.
pac_checkred

Value

data.frame.

Note

Results are cached for 30 minutes 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

```r
## Not run:
pacs::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 30 minutes 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

```r
## Not run:
pacs::pac_checkred("dplyr")
pacs::pac_checkred("dplyr", scope = c("ERROR"))
pacs::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)
```

Description

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

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pac</td>
<td>character a package name.</td>
</tr>
<tr>
<td>old</td>
<td>character an old version of package.</td>
</tr>
<tr>
<td>new</td>
<td>character a new version of package.</td>
</tr>
<tr>
<td>lib.loc</td>
<td>character vector. Default:</td>
</tr>
<tr>
<td>repos</td>
<td>character vector URLs of the repositories to use. Used only for the validation. Default <a href="https://cran.rstudio.com/">https://cran.rstudio.com/</a></td>
</tr>
</tbody>
</table>

Value

list with c("imports", "exports", "exportPatterns", "importClasses", "importMethods", "exportClasses", "exportMethods", "exportClassPatterns", "dynlibs", "S3methods") slots, and added and removed ones for each of them.
pac_compare_versions

Examples

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

```r
pac_compare_versions(
  pac,
  old = NULL,
  new = NULL,
  fields = c("Imports", "Depends", "LinkingTo"),
  lib.loc = .libPaths(),
  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** a character vector listing the types of dependencies, a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: c("Depends", "Imports", "LinkingTo")
- **lib.loc** character vector. Default: .libPaths()
- **repos** character vector URLs of the repositories to use. Used only for the validation. Default https://cran.rstudio.com/
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 both versions have the same requirement. -1 means that the new version remove this requirement. 1 means that the new version added a new requirement.

Examples

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

```r
pac_deps(
  pac,
  fields = c("Depends", "Imports", "LinkingTo"),
  lib.loc = .libPaths(),
  base = FALSE,
  local = TRUE,
  description_v = FALSE,
  attr = TRUE,
  recursive = TRUE,
  repos = biocran_repos()
)
```
pac_deps

Arguments

pac  character a package name.
fields a character vector listing the types of dependencies, a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: c("Depends", "Imports", "LinkingTo")
lib.loc character vector, used optionally when local is equal TRUE. Default: .libPaths()
base logical if to add base packages too. If TRUE then pacs::pacs_base() are taken into account. Default: FALSE
local logical if to use local repository or newest CRAN packages, where by default local packages are used. Default: TRUE
description_v if the dependencies version should be taken from description files, minimal required. By default installed versions are taken. Default: FALSE
attr logical specify if a package and its version should be added as a attribute of data.frame or for FALSE as an additional record. Default: TRUE
recursive logical If to assess the dependencies recursively. Default: TRUE
repos character vector URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos()

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 results could be aggregated like, stats::aggregate(results[, c("Version"), drop = 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)
```
Description

A higher-level function, build from pacs::pacs_deps. Package dependencies installed when e.g. R CMD check a package. "Depends", "Imports", "LinkingTo", "Suggests" fields from the DESCRIPTION file and their recursive dependencies taken from "Depends", "Imports", "LinkingTo" fields. Dependencies are taken remotely for the newest version.

Usage

```r
pac_deps_dev(
  pac,
  base = FALSE,
  local = FALSE,
  attr = TRUE,
  repos = pacs::biocran_repos()
)
```

Arguments

- **pac**: character a package name.
- **base**: logical if to add base packages too.
- **local**: logical if to use local repository or newest CRAN packages, where by default local packages are used. Default: TRUE
- **attr**: logical specify if a package and its version should be added as a attribute of data.frame or for FALSE as an additional record. Default: TRUE
- **repos**: character vector URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos() If TRUE then pacs::pacs_base() are taken into account. Default: FALSE

Examples

```r
## Not run:
pacs::pac_deps_dev("dplyr")
pacs::pac_deps_dev("pacs")
# with the main package in the list
pacs::pac_deps_dev("pacs", attr = 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 a character vector listing the types of dependencies, a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. 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)
Description

A higher-level function, build from pacs::pacs_deps. Package dependencies installed when run installed.packages. "Depends", "Imports", "LinkingTo" fields from the DESCRIPTION file and their recursive dependencies taken from the same fields. Dependencies are taken remotely for the newest version.

Usage

```r
pac_deps_user(
  pac,
  base = FALSE,
  local = FALSE,
  attr = TRUE,
  repos = pacs::biocran_repos()
)
```

Arguments

- `pac` character a package name.
- `base` logical if to add base packages too.
- `local` logical if to use local repository or newest CRAN packages, where by default local packages are used. Default: TRUE
- `attr` logical specify if a package and its version should be added as a attribute of data.frame or for FALSE as an additional record. Default: TRUE
- `repos` character vector URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos() If TRUE then pacs::pacs_base() are taken into account. Default: FALSE

Examples

```r
## Not run:
pacs::pac_deps_user("dplyr")
pacs::pac_deps_user("pacs")
# with the main package in the list
pacs::pac_deps_user("pacs", attr = FALSE)
## End(Not run)
```
pac_description

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 = .libPaths(), repos = "https://cran.rstudio.com/"
)

Arguments

pac  character a package name.

version  character package version, by default the is_installed version. Default: NULL

at  Date. Default: NULL

local  logical if to use local library. Default: FALSE

lib.loc  character vector, used optionally when local is equal TRUE. Default: .libPaths()

repos  character vector URLs of the repositories 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 30 minutes with memoise package.

Examples

## Not run:
pacs::pac_description("dplyr", version = "0.8.0")
pacs::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

A package health for a certain version or at a specific Date. A healthy package was published for more than \( x \) days, where default is 14 days. CRAN team gives around one/two week to resolve a package which gave errors under the check page. The newest release is checked for any warnings/errors on the R CRAN package check page.

Usage

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

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, \( \geq \text{limit} \). 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: .libPaths()
- **repos**: character vector URLs of the repositories to use. Default https://cran.rstudio.com/
- **source**: character one of c("crandb", "cran"). Using the MEATCRAN DB or the direct web page download from CRAN. Default: "crandb"

Value

logical if a package is healthy.
pac_isin

Note

Results are cached for 30 minutes with memoise package. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/. For source = "cran"the 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.

Examples

## Not run:
pacs::pac_health("memoise")
pacs::pac_health("dplyr", version = "0.8.0", limit = 14)
pacs::pac_health("dplyr", at = as.Date("2019-02-14"))
pacs::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 30 minutes 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 = .libPaths(), 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, it is omitted for non NULL version. Default: .libPaths()
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 30 minutes 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

```r
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 `pacsc::biocran_repos`.

Value

character most recent package version.

Note

Results are cached for 30 minutes with `memoise` package. For Bioconductor the newest one per R version.

Examples

```r
## 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

a package life duration for a certain version or at a specific Date.
**Usage**

```r
pac_lifeduration(
  pac,
  version = NULL,
  at = NULL,
  lib.loc = .libPaths(),
  repos = biocran_repos(),
  source = c("crandb", "cran")
)
```

**Arguments**

- **pac**: character a package name.
- **version**: character package version. By default the newest version is taken. The local repository has priority, it version is available. Default: NULL
- **at**: Date old version of package. Default: NULL
- **lib.loc**: character vector. Is omitted for non NULL version. Default: .libPaths()
- **repos**: character vector URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default pacs::biocran_repos()
- **source**: character one of c("crandb", "cran"). Using the MEATCRAN CRANDB or the direct web page download from CRAN. Default: "crandb"

**Value**

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

**Note**

Results are cached for 30 minutes with memoise package. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/. For source = "cran" the 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.

**Examples**

```r
# Not run:
pacs::pac_lifeduration("memoise")
pacs::pac_lifeduration("memoise", source = "cran")
pacs::pac_lifeduration("dplyr", version = "0.8.0")
pacs::pac_lifeduration("dplyr", at = as.Date("2019-02-14"))
# For Bioconductor packages it will work only for the newest per R version and installed ones.
pacs::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 = .libPaths(),
  repos = "https://cran.rstudio.com/"
)

Arguments

pac character a package name.
version character package version. By default the is_installed version. Default: NULL
at Date. Default: NULL
local logical if to use local library. Default: FALSE
lib.loc character vector, used optionally when local is equal TRUE. Default: .libPaths()
repos character vector URLs of the repositories 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 30 minutes with memoise package. This function is mainly built under source code from base::parseNamespaceFile.

Examples

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

## End(Not run)
pac_size

Size of the package

Description
Size of package.

Usage
pac_size(pac, lib.loc = .libPaths())

Arguments
pac character a package name.
lib.loc character vector. Default: .libPaths()

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,
  source = c("crandb", "cran")
)

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
source character one of c("crandb", "cran"). Using the METACRAN DB or the direct web page download from CRAN. Default: "crandb"

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. The base part of URL in the result is https://cran.r-project.org/src/contrib/.
Size character size of the tar.gz file.

Note

Results are cached for 30 minutes with memoise package. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/. For source = "cran" the 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.

Examples

## Not run:
pacs::pac_timemachine("dplyr", at = as.Date("2017-02-02"))
pacs::pac_timemachine("dplyr", from = as.Date("2017-02-02"), to = as.Date("2018-04-02"))
pacs::pac_timemachine("dplyr", at = Sys.Date())
pacs::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 all dependencies, recursively.

Usage

pac_true_size(
Pac,  
fields = c("Depends", "Imports", "LinkingTo"),  
lib.loc = .libPaths(),  
exclude_joint = 0L
)

Arguments

pac character a package name.
fields a character vector listing the types of dependencies, a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: c("Depends", "Imports", "LinkingTo")
lib.loc character vector. Default: .libPaths()
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. The default value of fields should be suited for almost all scenarios.

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

```r
pac_validate(
  pac,
  lib.loc = .libPaths(),
  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 vector. Default: `.libPaths()`
- **fields**: a character vector listing the types of dependencies, a subset of c("Depends", "Imports", "LinkingTo"). Character string "all" is shorthand for that vector, character string "most" for the same vector without "Enhances", character string "strong" (default) for the first three elements of that vector. Default: c("Depends", "Imports", "LinkingTo")
- **lifeduration**: logical if to assess life duration for each package in the library. MEATCRAN CRANDB is used for less than 500 packages. The direct web page download from CRAN or local evaluation for newest packages otherwise. 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 URLs of the repositories to use. By default checking CRAN and newest Bioconductor per R version. Default `pacs::biocran_repos()`

Value

data.frame with 5/7/8/9 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 dependency layer, direct dependencies from DESCRIPTION file.

**newest** logical (Internet needed) if the installed version is the newest one.

**cran** logical (Internet needed) if the package is on CRAN, version is not taken into account here.

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

**lifeduration** (Optional) (Internet needed) 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 when there is more than 500 packages. 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. The crandb R packages database is a part of METACRAN project, source: Csárdi G, Salmon M (2022). pkgsearch: Search and Query CRAN R Packages. https://github.com/r-hub/pkgsearch, https://r-hub.github.io/pkgsearch/.

**Examples**

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