Package 'RWsearch'

July 31, 2025

Title Lazy Search in R Packages, Task Views, CRAN, the Web. All-in-One Download

Description Search by keywords in R packages, task views, CRAN, the web and display the results in the console or in txt, html or pdf files. Download the package documentation (html index, README, NEWS, pdf manual, vignettes, source code, binaries) with a single instruction. Visualize the package dependencies and CRAN checks. Compare the package versions, unload and install the packages and their dependencies in a safe order. Explore CRAN archives. Use the above functions for task view maintenance. Access web search engines from the console thanks to 80+ bookmarks. All functions accept standard and non-standard evaluation.

Version 5.2.6

Date 2025-07-31

Depends R (>= 4.0.0)

Imports brew, latexpdf, networkD3, sig, sos, XML

Suggests ctv, findR, foghorn, pacman, litedown, xfun

Author Patrice Kiener [aut, cre] (ORCID: <https://orcid.org/0000-0002-0505-9920>)

Maintainer Patrice Kiener <rpackages@inmodelia.com>

License GPL-2

Encoding UTF-8

LazyData false

NeedsCompilation no

Language en-GB

VignetteBuilder litedown

RoxygenNote 7.3.2

Repository CRAN

Date/Publication 2025-07-31 12:10:02 UTC

Contents

RWsearch-package *Package RWsearch*

Description

Search by keywords in R packages, task views, CRAN, the web and display the results in the console or in txt, html or pdf files. Download the package documentation (html index, README, NEWS, pdf manual, vignettes, source code, binaries) with a single instruction. Visualize the package dependencies and CRAN checks. Compare the package versions, unload and install the packages and their dependencies in a safe order. Explore CRAN archives. Use the above functions for task view maintenance. Access web search engines from the console thanks to 80+ bookmarks. All functions accept standard and non-standard evaluation. Inspired by the packages ctv, foghorn, latexpdf, pacman and sos.

Author(s)

Maintainer: Patrice Kiener <rpackages@inmodelia.com> (ORCID)

Examples

```
### THE W IN RWsearch: LAUNCH WEBSITES AND SEARCH ENGINES
if (interactive()) {
h_cranbydate(repos = "https://cloud.r-project.org")
h_yt("Serge Gainsbourg Ne dis rien")
h_osm("La Ferriere sous Jougne")
h_mw(recension)
h_lexilogos()
}

### A CONVENIENT FUNCTION FOR NON-STANDARD EVALUATION
## Non-standard content (nsc1, nsc2), standard content ("stc3", "double word4")
## and regular object (obj) stored in .GlobalEnv can be merged with cnsc()
obj <- c("obj5", "obj6")
cnsc(nsc1, nsc2, "stc3", "double word4", obj)

### DOWNLOAD CRANDB, BINARYDB AND CHECKDB
## In real life, download crandb, binarydb and checkdb from CRAN or load them
## with functions crandb_down(), binarydb_down(), checkdb_down() or
## crandb_load(), binarydb_load() and checkdb_load().
## checkdb can be ignored if less than npkgs are explored.
## In this example, we use three small files.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
binarydb_load(system.file("data", "zbinarydb.rda", package = "RWsearch"))
checkdb_load(system.file("aabb", "zcheck_results.rds", package = "RWsearch"))

### PRINT THE LATEST PACKAGES UPLOADED ON CRAN (LAST DATE = "2024-09-20")
crandb_fromto(from = -30, to = max(crandb$Published))
crandb_fromto(from = "2024-01-01", to = Sys.Date())
```

```
### SEARCH IN CRANDB
## Search in crandb. Use standard or non-standard content.
## Display the results in a vector or in a list.
s_crandb(search, find, cran, web)
s_crandb(search, find, cran, web, select = "PD", mode = "and")
s_crandb("^f", select = "P")
s_crandb(c("thermodynamic", "chemical reaction"))
(lst <- s_crandb_list(thermodynamic, "chemical reaction"))
p_vers(lst)

### DISPLAY THE RESULTS
## in the console, in (txt, md, pdf) files or in the browser.
if (interactive()) {
p_table2(lst)
p_table7pdf(lst, dir = file.path(tempdir(), "ptable"), openpdf = TRUE)
p_text(lst, dir = file.path(tempdir(), "ptext1"), pager = TRUE,
           repos = "https://cloud.r-project.org")
p_text2pdf(lst, dir = file.path(tempdir(), "ptext1"), openpdf = TRUE,
           repos = "https://cloud.r-project.org")
p_display(lst, dir = file.path(tempdir(), "pdisp1"))
}

### VISUALIZE THE DOCUMENTATION (IN A BROWSER)
## from the installed packages on your computer
## or from the packages listed in https://search.r-project.org
if (interactive()) {
p_html(brew, sig)
p_man(foghorn)
p_pdfweb(sos, repos = "https://cloud.r-project.org")
}

### VISUALIZE THE PACKAGE DEPENDENCIES AND THE PACKAGE STATUS
if (interactive()) {
p_graphF(actuar, fitdistrplus, reverse = TRUE) # Children
p_graphF(RWsearch)    # Parents
p_vers(RWsearch)      # Installed versions
p_vers_deps(RWsearch) # Installed versions
}

### VISUALIZE THE PACKAGE CHECKS (USE checkdb FOR FASTER RESULTS)
if (interactive()) {
p_check(RWsearch, repos = "https://cloud.r-project.org")
res <- p_checkdeps_lst(RWsearch, repos = "https://cloud.r-project.org")
head(res, 3)
}

### DOWNLOAD THE DOCUMENTATION
## Vector => download in the "docpkgs" directory ("." is the current directory)
## List   => download in subdirectories named after the keywords
## (non-standard content is accepted)

p_down(pacman, pdfsearch, sos, dir = file.path(tempdir(), "pdown"),
       repos = "https://cloud.r-project.org")
```

```
p_down(lst, dir = file.path(tempdir(), "pdown"), repos = "https://cloud.r-project.org")


### SEARCH WITH sos (ON R-PROJECT HELP PAGES), rdrr AND rdoc
if (interactive()) {
(res <- s_sos(aluminium))
head(data.frame(res), 3)
tail(data.frame(res), 3)
h_rdrr(aluminium)
h_rdoc(distillation)
}

### TASK VIEW MAINTENANCE
## In real life, download crandb and tvdb from CRAN or load them from your directory
## with functions crandb_down(), tvdb_down(), crandb_load(), tvdb_load().
## In this example, we use small files.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
tvdb_load(system.file("data", "ztvdb.rda", package = "RWsearch"))

## List the task views
tvdb_vec()
tvdb_pkgs(ChemPhys, Robust)

## Search for some packages in the task views
s_tvdb(actuar, FatTailsR, MASS, zoo, nopackage)

## Search for recent packages in crandb that contain the keyword
## and verify if the packages are already referred in the task view.
## from = "2024-01-01" and "2018-01-01" are selected for this small example.
s_crandb_tvdb("distribution", tv = "Distributions", from = "2024-01-01")
s_crandb_tvdb("distribution", tv = "Distributions", from = "2018-01-01")

### EXPLORE CRAN ARCHIVE AND DOWNLOAD OLD tar.gz FILES
## In real life, download archivedb and crandb from CRAN
## with the functions archivedb_down() and crandb_down().
## In this example, we load two small files (94 and 85 packages).
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
archivedb_load(system.file("aabb", "zCRAN-archive.html", package = "RWsearch"))
archivedb_npkgs()
lapply(archivedb_list(), tail)

## Download the latest tar.gz version from CRAN archive
## (this works for both both existing and removed packages).

p_vers(fitur, zmatrix)   # zmatrix is an archived package
p_downarch(fitur, zmatrix, dir = file.path(tempdir(), "pdownarch"))
```

---

archivedb *CRAN archive (CRAN-archive.html + archivedb)*

Description

The following functions deal with the packages archived in CRAN. The html file downloaded from CRAN contains the regular packages that have been updated once and the packages that have been removed from CRAN by CRAN administrators. It does not contain the first version of the packages uploaded to CRAN and never updated. These files and the files removed from CRAN index can be guessed through a comparison with `crandb`.

`archivedb_down` downloads from CRAN the html file of the archived packages, saves it on the disk under the name `filename`, extracts from it and loads in .GlobalEnv a data.frame named `archivedb`.

`archivedb_load` reads the html file `filename` saved on the disk, extracts from it and loads in .GlobalEnv a data.frame named `archivedb`.

`archivedb_npkgs` returns the number of packages listed each category: number of packages in `crandb`, in `archivedb`, at first version, at subsequent version and removed from `crandb` (CRAN index).

`archivedb_pkgs` returns the packages listed in CRAN archive (= `archivedb`).

`archivedb_rempkgs` returns the packages removed from CRAN but available in CRAN archive. The result can be combined with `p_check` to display the last CRAN check performed (if available). See the example.

`archivedb_list` compares the data.frame `archivedb` and `crandb` and returns a list with the following items:

- pkgs_crandb: the packages listed in `crandb`.
- pkgs_archivedb: the packages listed in `archivedb`.
- pkgs_first: the packages in first version in `crandb`.
- pkgs_updated: the packages with more than one version in `crandb`.
- pkgs_removed: the archived packages removed from CRAN regular index, i.e. not listed in `crandb`.
- dfr_crandb: data.frame pkgs_crandb + Published date.
- dfr_archivedb: data.frame pkgs_archivedb + Archived date.
- dfr_first: data.frame pkgs_first + Published date.
- dfr_updated: data.frame pkgs_updated + Published date.
- dfr_removed: data.frame pkgs_removed+ Archived date.
- npkgs: the number of packages in each category.

Use `p_archive_lst` to list the package versions stored in CRAN archive.

Use `p_downarch` to download packages from CRAN archive, either the latest version or a specific version number.

Usage

```
archivedb_down(filename = "CRAN-archive.html", dir = ".",
  url = "https://cran.r-project.org/src/contrib/Archive")

archivedb_load(filename = "CRAN-archive.html")
```

```
archivedb_npkgs(archivedb = get("archivedb", envir = .GlobalEnv),
  crandb = get("crandb", envir = .GlobalEnv))

archivedb_pkgs(archivedb = get("archivedb", envir = .GlobalEnv))

archivedb_rempkgs(archivedb = get("archivedb", envir = .GlobalEnv),
  crandb = get("crandb", envir = .GlobalEnv))

archivedb_list(archivedb = get("archivedb", envir = .GlobalEnv),
  crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

filename	character. The path to file "CRAN-archive.html" (or equivalent).
dir	character. The directory where filename or tar.gz files are saved. Default value "." is the current directory.
url	character. The url address of CRAN archive html file.
archivedb	data.frame archivedb. The archivedb data.frame format loaded in memory by archivedb_down or archivedb_load.
crandb	data.frame crandb. The data.frame of CRAN packages.

Examples

```
### DOWNLOAD archivedb AND COMPARE IT WITH crandb.
## In real life, download archivedb and crandb from CRAN
## with the functions archivedb_down() and crandb_down().
## In this example, we load two small files made with 94 and 85 packages.

crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
archivedb_load(system.file("aabb", "zCRAN-archive.html", package = "RWsearch"))
archivedb_npkgs()
archivedb_pkgs()
archivedb_rempkgs()

lst <- archivedb_list()
lapply(lst, head)
lapply(lst, tail)

xlim <- as.Date(range(lst$dfr_archivedb$Archived)) ; xlim
op <- par(mfrow = c(2,1))
hist(as.Date(lst$dfr_first$Published),
     breaks = 12, freq = TRUE, las = 1, xlim = xlim)
hist(as.Date(lst$dfr_archivedb$Archived),
     breaks = 12, freq = TRUE, las = 1, xlim = xlim)
par(op)
```

binarydb *CRAN Matrix Of Available Binary Packages (archivedb.rda)*

Description

binarydb_down downloads the matrix of the packages available in binary format available in CRAN (Windows or macOS depending the computer), saves it it in a file named binarydb and finally loads it in .GlobalEnv under the name binarydb. It is a wrapper around the function available.packages(type = "binary").

binarydb_load loads the file filename in .GlobalEnv under the name binarydb.

When loaded in .GlobalEnv, binarydb is recognized by the functions p_vers and p_vers_deps to compare the package version numbers of the installed packages with the most recent versions of the binary and the source versions available in CRAN.

Usage

```
binarydb_down(dir = ".", repos = getOption("repos")[1])

binarydb_load(filename = "binarydb.rda")
```

Arguments

dir	character. The directory where "binarydb.rda" is saved Default value "." is the current directory.
repos	character. The address of your local CRAN.
filename	character. The (path to a) file "binarydb.rda" or an equivalent.

Examples

```
## In real life, to get the latest version, use
## binarydb_down()
## Here, we use a small file
binarydb_load(system.file("data", "zbinarydb.rda", package = "RWsearch"))
class(binarydb)
colnames(binarydb)
dim(binarydb)
head(binarydb)
```

checkdb *CRAN checks file (check_results.rds)*

Description

checkdb_down downloads from CRAN the file *check_results.rds*, saves it unchanged in the designated directory and loads it in .GlobalEnv under the name checkdb.

checkdb_load loads the file check_results.rds in .GlobalEnv under the name checkdb.

checkdb is a data.frame of dimension 200000 x 10 (approximatively). It is used by the functions p_check_lst and p_checkdeps_lst.

Usage

```
checkdb_down(dir = ".", repos = getOption("repos")[1])

checkdb_load(filename = "check_results.rds")
```

Arguments

dir	character. The directory where filename or tar.gz files are saved. Default value "." is the current directory.
repos	character. The address of your local CRAN.
filename	character. The path to file "check_results.rds" (or equivalent).

Examples

```
### zcheck_results.rds is a subset of 94 packages synchronized with zcrandb.

checkdb_load(system.file("aabb", "zcheck_results.rds", package = "RWsearch"))
unique(checkdb$Package)
dim(checkdb)
head(checkdb, 15)
```

cnsc *Combine Standard and Non-Standard Content*

Description

cnsc reads standard content (object in .GlobalEnv or quoted characters) and non-standard content. Standard content, including *call*s, is evaluated. Non-standard content and non-existing objects (in .GlobalEnv) are converted into character chains. Regular lists are kept unchanged.

cnscinfun is used inside cnsc. It should not be called directly. If you find cnsc and cnscinfun appealing, copy the code of cnscinfun in your package and use it as an hidden function.

Usage

```
cnsc(...)

cnscinfun()

cnscinfun2(...)
```

Arguments

...	Character vectors, standard or non-standard, existing or non-existing R objects, regular call. Examples : "word1"; c("word1 word2"); c("word1", "word2"); "word1", "word2", "word3"; word1, word2, c("word3", "word4").

Examples

```
### cnsc
## Non-standard content (nsc1, nsc2), standard content ("stc3", "double word4")
## and regular object (vec) stored in .GlobalEnv are merged.
vec <- c("obj5", "obj6")
cnsc(nsc1, nsc2, "stc3", "double word4", vec)

## Lists, either name in .GlobalEnv or call, are evaluated.
lst <- list(A = c("txt1","txt2","txt3"), B = c("txt4", "txt5"))
cnsc(lst)
cnsc(list(C = c("pkg1","pkg2","pkg3"), D = c("pkg4","pkg5")))

### cnscinfun
fun <- function(...) cnscinfun()
fun(nsc1, nsc2, "stc3", "double word4", vec)
fun(lst)

### cnscinfun used in RWsearch: one line at the begining of each function.
### An easy-to-use Non Standard Evaluation, mainly for characters.
funsort <- function(..., char = NULL) {
  words <- if (is.null(char)) cnscinfun() else char
  sort(words)
  # or more complex code
}
funsort(nsc1, nsc2, "stc3", "double word4", vec)
funsort(char = sample(vec, 5, replace = TRUE))
```

Description

crandb_down downloads from CRAN the file *packages.rds*, a file refreshed everyday that describes the packages available in CRAN for this day, opens it as a data.frame and cleans this data.frame with the following oprations: rename (with make.names) the column names that are syntactically invalid, remove the duplicated lines located at the end of the file, clean some bad characters in the Description column. The resulting clean data.frame is then loaded in .GlobalEnv under the name crandb and saved in the current directory with the filename crandb.rda. If oldfile is defined, the vector of packages between the two files is compared and a short message is printed about the differences (removed packages, new packages, updated packages).

crandb_load loads the file filename in .GlobalEnv under the name crandb. It embeds the function load("crandb.rda") and add a short message about the data.frame properties.

crandb_pkgs displays all packages listed in crandb. The number of packages is larger than the number obtained with nrow(available.packages()) since packages for all OSes are counted.

crandb_fromto displays the packages published in CRAN between two dates.

Usage

```
crandb_down(dir = ".", oldfile = "crandb.rda", verbose = TRUE,
  repos = getOption("repos")[1])

crandb_load(filename = "crandb.rda")

crandb_comp(filename = "crandb.rda", oldfile = "crandb-old.rda",
  addtxt = "")

crandb_pkgs(bydate = FALSE, rev = FALSE, crandb = get("crandb", envir =
  .GlobalEnv))

crandb_fromto(from = -10, to = Sys.Date(), crandb = get("crandb", envir =
  .GlobalEnv))
```

Arguments

dir	character. The directory where "crandb.rda" is saved and the old "crandb.rda" is read. Default value "." is the current directory.
oldfile	character or NULL. The (path to an) old file that will be compared to a freshly downloaded version of "crandb.rda" or to filename. Set to NULL if no comparison is required.
verbose	logical. TRUE prints the result. FALSE keeps it invisible.
repos	character. The address of your local CRAN.
filename	character. The (path to a) file "crandb.rda" or an equivalent.
addtxt	character. Internal use.
bydate	logical. List the package by date of publication rather than by alphabetical order.
rev	logical. Print in reverse order.
crandb	data.frame crandb. The data.frame of CRAN packages.

from	Negative integer or character representing a date. The number of days preceeding to or a date before to.
to	date. The upper date in the search.

Examples

```
### In this example, we use the small file zcrandb.rda made of 94 packages
## extracted from CRAN on 2024-09-27. In real life, use crandb_down().
# crandb_down(dir = tempdir(), repos = "https://cloud.r-project.org")

crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
crandb_pkgs()
dim(crandb)
colnames(crandb)
crandb$Published
crandb_fromto(from = "2024-01-01", to = Sys.Date())
pkgs <- crandb_fromto(from = -60, to = max(crandb$Published)) ; pkgs

## Print the table in the console (better if full width)
p_table2(pkgs)

## Display in the browser
if (interactive()) {
p_display7(pkgs, dir = file.path(tempdir(), "crandbdown"))
}


### In the real life, we use a fresh file downloaded from CRAN (6.4 Mo / 20").
## Here, we retrieve the packages uploaded during the last 2 days.
```

---

cranmirrors_down *CRAN archive (CRAN-archive.html + archivedb)*

Description

cranmirrors_down downloads the csv file of CRAN mirrors, modifies the "Maintainer" and "Host" columns, eventually saves the modified data.frame on the disk, loads this data.frame in .GlobalEnv and prints in the console a subset of the data.frame made of the selected columns.

Usage

```
cranmirrors_down(filename = "CRAN-mirrors1.csv", dir = ".",
  columns = c(1, 2, 3, 7), save = FALSE,
  url = "https://cran.r-project.org/CRAN_mirrors.csv")
```

Arguments

filename	character. The path to file "CRAN-mirrors1.csv" (or equivalent).
dir	character. The directory where filename is saved. Default value "." is the current directory.
columns	a vector of integers or a vector of names. The column numbers or the column names. Allowed numbers are within 1:9. Allowed names are: "Name", "Country", "City", "URL", "Host", "Maintainer", "OK", "CountryCode" and "Comment".
save	logical. Save the file. If FALSE, the default, the file is just loaded in .GlobalEnv and a subset is printed in the console.
url	character. The url address of the CRAN csv file.

Examples

```
cranmirrors_down(dir = file.path(tempdir(), "cranmirrors"), save = TRUE)
```

---

e_check *Check Results of Packages Identified by their Email Address*

Description

e_check opens the browser and returns the "CRAN Check Results" page(s) of the packages maintained by one or several maintainers identified by their regular email addresses (but not the orphaned ones). An internet connection is required. This function is a simplified version of the functions proposed in the package *foghorn*.

Usage

```
e_check(..., char = NULL, repos = getOption("repos")[1])
```

Arguments

...	any format recognized by cnsc, except list. A vector of quoted "e-mail adresses".
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
repos	character. The address of your local CRAN.

Examples

```
if (interactive()) {
e_check(c("rpackages@inmodelia.com", "christophe.dutang@ensimag.fr"),
        repos = "https://cloud.r-project.org")
}
```

funmaintext *Modify the Main Text and the Markdown Header in p_text Function*

Description

Use funmaintext or funmaintext2 to select the function that displays the main text. Usage is f_maintext = funmaintext (without curly braces).

Use funheadermd to insert markdown header in function sep1 = funheadermd() (with curly braces).

See the example in p_text. To create you own functions, use these functions as a pattern. The five parameters in f_maintext, funmaintext are mandatory. funheadermd can be freely modified.

Usage

```
funmaintext(pkg, sep1, sep2, eol, crandb, repos)

funmaintex(pkg, sep1, sep2, eol, crandb, repos)

funheadermd(title = "TITLE", author = "AUTHOR", date = Sys.Date(),
  keep_tex = "false", toc = "false", number_sections = "true",
  fontsize = "10pt", papersize = "a4paper", margin = "1in")

funheadertex(fontsize = "10pt", papersize = "a4paper", margin = "1in")

funfootertex()
```

Arguments

pkg	character. The package name.
sep1	character. The symbols written just before each package name.
sep2	character. The symbols written just after each package name.
eol	character. The end of line for the main text (but not for the header and the footer). "\n" for text, " \n" for rmarkdown, " \\ \n" for latex.
crandb	data.frame crandb. The data.frame of CRAN packages.
repos	character. The address of your local CRAN.
title	character. The title of the .md document (and then in the .pdf file).
author	character. The author of the .md document.
date	character. The date of the document. Any text format is accepted.
keep_tex	character. "true" or "false".
toc	character. "true" or "false".
number_sections	character. "true" or "false".
fontsize	character. Usually "10pt", "11pt", "12pt.
papersize	character. The usual tex format. Example: "a4paper".
margin	character. In inches, cm or mm. Example: "0.5in", "1.5cm", "25mm".

f_args *Names and Arguments of Functions*

Description

f_args is a wrapper of the base function args.

f_sig prints the name and arguments of one or several functions in a readable style. It wraps the function sig::sig.

p_sig prints the name and arguments of the functions exported by one or several packages. It wraps the function sig::list_sigs.

Usage

```
f_args(..., char = NULL)

f_sig(..., char = NULL)

p_sig(..., char = NULL, pattern = NULL)
```

Arguments

...	any format recognized by cnsc, except list. A vector of unquoted "functions" or characters.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
pattern	a regular expression. See the example.

Examples

```
f_args(mean, p_display)
f_sig( mean, p_display)

library(brew)
library(sos)
p_sig(brew)
p_sig(RWsearch, sos, pattern = "^f")
```

f_pdf *PDF Pages of Functions*

Description

f_pdf generates in the current directory the pdf pages of one or several functions. The pdf pages are printed but not opened. Miktex or Texlive is required. This function wraps {utils::help} with the pdf option activated. Similar functions are available in the packages *document* and *sinew*.

Usage

```
f_pdf(..., char = NULL)
```

Arguments

...	any format recognized by cnsc, except list. A vector of quoted "package::function".
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.

Examples

```
## FALSE is here to avoid a NOTE in CRAN checks. Ignore this line.
if (FALSE) {
   f_pdf(c("RWsearch::cnsc", "RWsearch::p_inun"))
}
```

h_direct *Open a Web Page in the your browser*

Description

h_framasoft gives access to several free web services (as in speech and in half pint of beer) that are good alternatives to GAFA services.

h_academie is a bookmark to the Academie francaise's dictionary.

h_lexilogos gives access to hundreds of dictionaries in many languages.

h_deepl, h_googletranslate, h_interglot, h_reverso, h_linguee, h_promt, h_reverso, h_systran provide translation engines. h_linguee returns examples with long sentences.

h_yacy is a decentralized peer-to-peer web search software.

h_etz is a bookmark to the EveryTimeZone website.

h_tad and h_tadsm are bookmarks to timeanddate.com, a website dedicated to date and time conversion plus timezone management.

h_meteoblue and h_windy are bookmarks to the Meteoblue and Windy meteo and weather websites.

Usage

```
h_academie()

h_deepl()

h_etz()

h_framasoft()

h_framasoft0()

h_googletranslate()

h_interglot()

h_lexilogos()

h_linguee()

h_meteoblue()

h_promt()

h_reverso()

h_systran()

h_tad()

h_tadsm()

h_windy()

h_yacy()
```

Examples

```
if (interactive()) {
h_linguee()
h_lexilogos()
}
```

---

h_engine *Explore the Web with Various Search Engines*

Description

Launch the default browser and search in: ABC Bourse (short stock names), arXiv (vectorized), Ask, Baidu, Blackle, Bing, Bing Map (bmap), Boursorama (short stocknames), CNRTL (French dictionary), Collins English Dictionary, CPAN and metaCPAN (Perl), Crossref (DOI and bibliographic metadata), CTAN (Latex), Daum, DailyMotion (dm), DOI, DuckDuckGo (ddg), Ecosia, Egerin, Evene (citations), Exalead, Excite, Gigablast, GitHub, GitLab, Google Map (gmap), Google, Google Scholar (gscholar), IANA TLD root domain database, IANA WHOIS service, Info, Khoj, Les Echos, La Tribune (lt), Lilo, Lycos, Mappy Map, Merriam-Webster (mw, English dictionary), Nabble, Nate, Naver (see N2H4 package), Orcid, Open Street Map, OSM Nominatim, Parsijoo, PeerTube, Peru, Pipilika, Qwant (qw + qwfr), R-bloggers, Rdocumentation (rdoc), Rdocumentation task views (rdoctv), Rdrr, Reverso dictionnary, Rseek, Sapo, Searx, Sogou, SSRN and SSRN Author (vectorized), Stackoverflow (so), Startpage (ex-Ixquick), Twitter (+ twfr), L'Usine Nouvelle (un), ViaMichelin Map and Routes (via), Les Verbes, Vimeo, Wego (Here maps), Wikipedia (wp + wpfr), Yahoo, Yahoo Finance, Yandex, Yooz, Youtube (yt).

`h_zbib` is a bookmark to ZoteroBib, a service that returns the complete bibliographic reference from a fragment of information: URL, ISBN, DOI, PMID, arXiv id or title and generates a MD5 number to retrieve it later.t

Using the regular R format "w1 w2 w3" rather than w1, w2, w3 makes sense as most functions collapse the words into character chains "w1 w2 w3", "w1+w2+w3" or "w1-w2-w3".

Visit https://en.wikipedia.org/wiki/Web_search_engine for a list of web search engines.

Usage

```
h_abcbourse(..., char = NULL)

h_ask(..., char = NULL)

h_arxiv(..., char = NULL)

h_arxivpdf(..., char = NULL)

h_baidu(..., char = NULL)

h_blackle(..., char = NULL)

h_bing(..., char = NULL)

h_biorxiv(..., char = NULL)

h_biorxivpdf(..., char = NULL)

h_bmap(..., char = NULL)

h_boursorama(..., char = NULL)

h_cnrtl(..., char = NULL)
```

```
h_collins(..., char = NULL)

h_cpan(..., char = NULL)

h_crossref(..., char = NULL)

h_ctan(..., char = NULL)

h_daum(..., char = NULL)

h_ddg(..., char = NULL)

h_dm(..., char = NULL)

h_doi(..., char = NULL)

h_ecosia(..., char = NULL)

h_egerin(..., char = NULL)

h_estrep(..., char = NULL)

h_evene(..., char = NULL)

h_exalead(..., char = NULL)

h_excite(..., char = NULL)

h_framabee(..., char = NULL)

h_gigablast(..., char = NULL)

h_github(..., char = NULL)

h_gitlab(..., char = NULL)

h_gmap(..., char = NULL)

h_google(..., char = NULL)

h_gscholar(..., char = NULL)

h_ianaTLD(..., char = NULL)

h_ianaWHOIS(..., char = NULL)

h_info(..., char = NULL)
```

```
h_ixquick(..., char = NULL)

h_khoj(..., char = NULL)

h_lesechos(..., char = NULL)

h_lilo(..., char = NULL)

h_lt(..., char = NULL)

h_lycos(..., char = NULL)

h_mappy(..., char = NULL)

h_mw(..., char = NULL)

h_nate(..., char = NULL)

h_naver(..., char = NULL)

h_orcid(..., char = NULL)

h_osm(..., char = NULL)

h_osmn(..., char = NULL)

h_parsijoo(..., char = NULL)

h_peertube(..., char = NULL)

h_peru(..., char = NULL)

h_pipilika(..., char = NULL)

h_qwant(..., char = NULL, lang = "en")

h_qwfr(..., char = NULL)

h_reverso_d(..., char = NULL)

h_sapo(..., char = NULL)

h_searx(..., char = NULL)

h_so(..., char = NULL)

h_sogou(..., char = NULL)
```

```
h_ssrn(..., char = NULL)

h_ssrnauth(..., char = NULL)

h_startpage(..., char = NULL)

h_twfr(..., char = NULL)

h_twitter(..., char = NULL, lang = "en")

h_un(..., char = NULL)

h_verbes(..., char = NULL)

h_via(..., char = NULL)

h_vimeo(..., char = NULL)

h_wego(..., char = NULL)

h_wp(..., char = NULL, lang = "en")

h_wpfr(..., char = NULL)

h_yahoo(..., char = NULL, lang = "en")

h_yahoofin(..., char = NULL, lang = "en")

h_yandex(..., char = NULL)

h_yooz(..., char = NULL)

h_yt(..., char = NULL)

h_zbib(..., char = NULL)
```

Arguments

...	any format recognized by cnsc, except list. A vector of packages.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function.
lang	character. The language accepted by the search engine, usually "en", "de", "es", "fr", "jp", etc.

Examples

```
if (interactive()) {
h_yt("Serge Gainsbourg Ne dis rien")
h_so(R, deep, neural, network)
```

```
h_osm("Le Chateau d'Oleron")
h_mw(recension)
h_arxiv(c(1212.4320, 1605.08732))
h_doi("10.1016/j.ejor.2013.06.029")
}
```

---

h_R *Open a Web Page in the Browser*

Description

h_R opens the page https://www.r-project.org.

h_Rml opens the page dedicated to the mailing lists https://www.r-project.org/mail.html.

h_Rnews opens the page https://cran.r-project.org/doc/manuals/r-devel/NEWS.html.

h_Rpkgs opens the page of the HTML documentation of base and CRAN packages (+ full list) https://search.r-project.org/R/doc/html/packages.html.

h_Rversions opens a page (from rversions package) that keeps a record of all R versions and their release dates.

h_cran opens the page of you local CRAN.

h_cranbydate and h_cranbyname open the page of CRAN packages sorted by date of publication and in alphabetical order.

h_cranchecks and h_crancheckwindows open the pages related to the checks of all packages listed by name, maintainers, dates, os. A special page is dedicated to Windows packages with the results for the previous, the current and the devel R versions.

h_crantv opens the page of CRAN task views.

h_cranberries, h_nabble, h_rbloggers, h_rdoc, h_rdoctv (RDocumentation), h_rdrr, h_rseek open the pages of web sites related to R.

h_gepuro lists all (most) R packages available on GitHub. A huge file.

Usage

```
h_R()

h_Rblog()

h_Rman()

h_Rml()

h_Rnews(repos = getOption("repos")[1])

h_Rpkgs()

h_Rversions(repos = getOption("repos")[1])
```

```
h_cran(repos = getOption("repos")[1])

h_cranbydate(repos = getOption("repos")[1])

h_cranbyname(repos = getOption("repos")[1])

h_cranchecks(repos = getOption("repos")[1])

h_crancheckwindows(repos = getOption("repos")[1])

h_crantv(repos = getOption("repos")[1])

h_cranstatus()

h_cranberries()

h_gepuro()

h_nabble(..., char = NULL)

h_rbloggers(..., char = NULL)

h_rdoc(..., char = NULL)

h_rdoctv(..., char = NULL)

h_rdrr(..., char = NULL)

h_rseek(..., char = NULL)

h_biocstats()
```

Arguments

repos	character. The address of your local CRAN.
...	any format recognized by cnsc, except list. A regular web address.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function.

Examples

```
if (interactive()) {
h_crantv(repos = "https://cloud.r-project.org")
h_cranberries()
}
```

h_ttp *Open a Web Page in the Browser*

Description

h_ttp opens the page coresponding to the mentionned address in the default browser.

Usage

```
h_ttp(..., char = NULL, https = TRUE, www = FALSE)
```

Arguments

...	any format recognized by cnsc, except list. A regular web address.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function.
https	logical. Use https or http.
www	logical. Add www. to the address.

Examples

```
if (interactive()) {
h_ttp("www.r-project.org")
}
```

p_archive *Read Packages in CRAN archive*

Description

p_archive opens in the browser one page per package and displays the package versions stored in CRAN archive.

p_archive_lst prints in the console a list of the package versions stored in CRAN archive.

l_targz takes as input the list issued by p_archive_lst and lists the last package versions archived before a certain date.

Use p_downarch to download packages from the CRAN archives, either the latest version stored or a specific version number.

Use archivedb_list to list all packages stored in CRAN archive (does not include the valid packages having a single version which are stored in regular CRAN).

Usage

```
p_archive(..., char = NULL)

p_archive_lst(..., char = NULL)

l_targz(lst, before = Sys.Date())
```

Arguments

...	any format recognized by cnsc, except list. A vector of packages.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
lst	list. A list produced by p_archive_lst.
before	character which can be converted to a Date, for instance "2017-05-14". Extract from CRAN archive the package(s) available before this date. Can be synchronized with the release dates of base-R versions listed at: https://CRAN.R-project.org/src/contrib/ and https://CRAN.R-project.org/package=rversions/readme/README.html

Examples

```
if (interactive()) p_archive(brew, RWsearch, NotAPkg)

lst <- p_archive_lst(RWsearch, zmatrix, NotAPkg) ; lst
l_targz(lst, before = "2020-01-01")
```

---

p_check *Return CRAN Package Check Results*

Description

p_check opens the default browser, connects to your local CRAN and displays for each package the CRAN Package Check Results or the last Check Results recorded in CRAN archive (with the date of the archive). An internet connexion is required.

p_check_lst reads the check results from the repository and print the results as a list in the console, with a message for the archived package(s). An internet connexion is required. If a large number of packages is to be analyzed, a preload of checkdb is required before launching the instruction (via checkdb_down or checkdb_load). This preload speeds up significantly the analysis.

p_checkdeps and p_checkdeps_lst extend the analysis to the package dependencies.

Comprehensive tables of the check results for package sources and Windows binaries can be displayed with h_cranchecks and h_crancheckwindows.

Usage

```
p_check(..., char = NULL, repos = getOption("repos")[1])

p_check_lst(..., char = NULL, npkgs = 10, repos = getOption("repos")[1])

p_checkdeps(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, crandb = get("crandb", envir = .GlobalEnv),
  repos = getOption("repos")[1])

p_checkdeps_lst(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, npkgs = 10, crandb = get("crandb", envir =
  .GlobalEnv), repos = getOption("repos")[1])
```

Arguments

`...`	any format recognized by cnsc, except list. A vector of packages.
`char`	(name to) a character vector. Use this argument if `...` fails or if you call the function from another function. If used, argument `...` is ignored.
`repos`	character. The address of your local CRAN.
`npkgs`	integer. The number of packages from which a preload of checkdb is required (via checkdb_down or checkdb_load).
`which`	character vector. A sub-vector of `c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")`. The short forms "D", "I", "L", "S", "N", "DL", "DI", "DIL", "DILS", "DILN", "DILSN", "SN" are accepted. "N" is for "Enhances" as the single letter "E" is used by R as a shortcut to EXPR, a reserved word.
`recursive`	logical. Search for (reverse) dependencies of (reverse) dependencies.
`reverse`	logical. Search for reverse dependencies.
`crandb`	data.frame `crandb`. Also accepted is `NULL` which will search in the local `installed.packages()`. This later form allows (private) packages that are not listed in `crandb`.

Examples

```
## In real life, download crandb and checkdb from CRAN or load them
## with functions crandb_down(), crandb_load(), checkdb_down(), checkdb_load().
## checkdb can be ignored if less than npkgs are explored.
## In these examples, we use two small files of 110 and 107 packages.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
checkdb_load(system.file("aabb", "zcheck_results.rds", package = "RWsearch"))

if (interactive()) {
p_check(RWsearch, igraph, zmatrix, NotApkg, repos = "https://cloud.r-project.org")
p_check_lst(RWsearch, igraph, zmatrix, NotApkg, repos = "https://cloud.r-project.org")
p_checkdeps_lst(igraph, zmatrix, NotApkg, repos = "https://cloud.r-project.org")
}
```

p_deadline *Packages Flagged by CRAN With a Deadline*

Description

p_deadline returns a table with the packages that failed to R CMD check and are flagged by CRAN to make some changes before a certain deadline. This feature has been introduced in the new version R_4.4.1 released mid-June 2024.

Usage

```
p_deadline(crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

crandb	data.frame crandb.

Examples

```
## In real life, download crandb with function crandb_down(), crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
dfr <- p_deadline() ; dfr
if (interactive() & nrow(dfr) > 0) {
p_check(dfr$Package, repos = "https://cloud.r-project.org")
}
```

p_deps *Dependencies and Reverse Dependencies of Packages*

Description

p_deps returns the (reverse) dependencies of a (vector of) package(s). It is a wrapper of the tools::package_dependencies function. A warning is issued for packages that are not in crandb + .libPaths() (for instance in CRAN archive, Bioconductor, Github or your own directories).

p_depsrec is a shortcut to p_deps(recursive = TRUE). It returns the recursive dependencies (e.g. the list of all ancestors).

p_depsrev is a shortcut to p_deps(reverse = TRUE). It returns the reverse dependencies (e.g. the children packages).

p_deps_deps returns a list with the recursive dependencies for the packages packages and every first level dependencies (including or excluding the ones in .Library.

p_deps_ndeps returns a vector of the number of dependencies for each package and their parent dependencies. With the argument sort = TRUE, the packages are listed from no dependency to the largest number of dependencies.

p_deps_count counts the number of (recursive/reverse) dependencies for each package and returns a data.frame with 4 columns: Parents1, ParentsN, Children1, ChildrenN.

p_deps_inpkgs returns the package dependencies that are installed in the computer.

p_deps_unpkgs returns the package dependencies that are not installed in the computer.

p_deps_inun combines p_deps and p_inun_crandb, then returns the status of all dependencies: installed or not installed in the computer, available or not available in the current crandb (see CRAN archives, Bioconductor, Github, your own packages).

The missing packages available on CRAN can be downloaded with p_down0, downloaded and checked (by R CMD check) with xfun::rev_check or installed with install.packages. The packages removed from CRAN but available in CRAN archive can be downloaded with p_downarch.

Usage

```
p_deps(..., char = NULL, which = "DIL", recursive = FALSE,
  reverse = FALSE, verbose = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))

p_depsrec(..., char = NULL, which = "DIL", reverse = FALSE,
  verbose = TRUE, crandb = get("crandb", envir = .GlobalEnv))

p_depsrev(..., char = NULL, which = "DIL", recursive = FALSE,
  verbose = TRUE, crandb = get("crandb", envir = .GlobalEnv))

p_deps_deps(..., char = NULL, which = "DIL", Library = FALSE,
  verbose = TRUE, crandb = get("crandb", envir = .GlobalEnv))

p_deps_ndeps(..., char = NULL, which = "DIL", Library = FALSE,
  sort = TRUE, verbose = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))

p_deps_count(..., char = NULL, which = "DIL", verbose = TRUE,
  crandb = get("crandb", envir = .GlobalEnv))

p_deps_inpkgs(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, crandb = get("crandb", envir = .GlobalEnv))

p_deps_unpkgs(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, crandb = get("crandb", envir = .GlobalEnv))

p_deps_inun(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

...	any format recognized by cnsc, excluding list. A package or a vector of packages listed in crandb or in installed.packages().
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.

which	character vector. A sub-vector of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). The short forms "D", "I", "L", "S", "N", "DL", "DI", "DIL", "DILS", "DILN", "DILSN", "SN" are accepted. "N" is for "Enhances" as the single letter "E" is used by R as a shortcut to EXPR, a reserved word.
recursive	logical. Search for (reverse) dependencies of (reverse) dependencies.
reverse	logical. Search for reverse dependencies.
verbose	logical. Print the message.
crandb	data.frame crandb. Also accepted is NULL which will search in the local installed.packages(). This later form allows (private) packages that are not listed in crandb.
Library	logical. The default FALSE excludes the base and recommended packages stored in .Library. TRUE includes them.
sort	logical. The default TRUE sorts the package by the number of dependencies.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

p_deps(networkD3, visNetwork)
p_deps(networkD3, visNetwork, recursive = TRUE)
p_deps(actuar, fitdistrplus, reverse = TRUE, which = "DILSN")

p_deps_inpkgs(RWsearch, canprot)
p_deps_unpkgs(RWsearch, canprot)
p_deps_inun(RWsearch, canprot, NotApkg)

p_deps_deps(actuar, networkD3, FatTailsR, RWsearch, NotApkg)
p_deps_ndeps(actuar, networkD3, FatTailsR, RWsearch, NotApkg)
p_deps_count(actuar, networkD3, FatTailsR, RWsearch, NotApkg)
```

---

p_display *Display Package Information in HTML Pages*

Description

p_display, p_display5 and p_display7 open the default browser and display the results of p_table, p_table5 and p_table7 in one or several html pages. If ... (or char) is a list, several pages are opened.

Usage

```
p_display(..., char = NULL, columns = c("Package", "Title", "Description"),
  dir = tempdir(), verbose = FALSE, crandb = get("crandb", envir =
  .GlobalEnv))

p_display5(..., char = NULL, dir = tempdir(), verbose = FALSE,
  crandb = get("crandb", envir = .GlobalEnv))

p_display7(..., char = NULL, dir = tempdir(), verbose = FALSE,
  crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

...	any format recognized by cnsc, including list. A vector or a list of packages. Or a vector or a list of data.frame produced by p_table.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
columns	character vector. A sub-vector of colnames(crandb). The short form "P", "T", "D", "PT", "PD", "TD", "PTD", "A", "M", "AM" describing the Package name, Title, Description, Author, Maintainer or a combination of them is accepted.
dir	character. The directory in which the html file(s) is (are) saved. tempdir() or getwd() are common paths.
verbose	logical. List the generated html file(s).
crandb	data.frame crandb. The data.frame of CRAN packages.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
## No package has the 'distillation' keyword. An empty table is returned.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
(lst <- s_crandb_list("thermodynamic", "chemical reaction", "distillation"))

if (interactive()) {
dir <- file.path(tempdir(), "pdisplay")

## Vector => 1 page
p_display(RWsearch, pacman, pdfsearch, sos, brew, dir = dir)

## List with 3 items => 3 pages with one empty
p_display5(lst, dir = dir)
}
```

p_down *Download the Package Documentation in One Directory or in Several Subdirectories*

Description

If pkgs is a vector of packages obtained from s_crandb, p_down downloads from CRAN and saves in the dir directory (by default the current directory) the index page, the manual, the vignettes, the README, NEWS, ChangeLog, CRAN checks files, the source code in *pkg_ver.tar.gz* format, the binary code in *pkg_ver.tgz* (Mac OSX) or *pkg_ver.zip* (Windows) format and a minimal R-script of each package. The files that do not exist are ignored, with no warning.

If pkgs is a list of packages obtained from s_crandb_list, p_down saves the downloaded files in subdirectories named after the names of the list, e.g. the keywords used at the search step. The names are eventually modified with gsub(".", "_", make.names(pkg), fixed = TRUE) to cope with Unix and Windows directory names.

p_down0 calls p_down with different values for each argument. With the default configuration, this function downloads nothing. It is mostly used to download one specific item which has not been previously downloaded.

p_downarch downloads from CRAN archive the tar.gz file of one or several packages, either the last version(s) with Sys.Date() or the version(s) before a given date. It combines 3 functions: p_archive_lst lists the packages stored in CRAN archive and their version numbers, l_targz extracts the appropriate version numbers available before a given date, targz_down downloads the tar.gz files in the selected directory (default is the current directory) and eventually decompresses it.

targz_down downloads the tar.gz files from CRAN archive to the selected directory (default is the current directory) and eventually decompresses it. If url = "https://cran.r-project.org/src/contrib", targz_down will take the latest version of the package.

Usage

```
p_down(..., char = NULL, index = TRUE, manual = TRUE, vignettes = TRUE,
  RqmdRmd = FALSE, README = TRUE, NEWS = FALSE, ChangeLog = FALSE,
  checks = FALSE, targz = FALSE, untar = FALSE, binary = FALSE,
  type = "binary", script = FALSE, dir = ".", crandb = get("crandb",
  envir = .GlobalEnv), repos = getOption("repos")[1])

p_down0(..., char = NULL, index = FALSE, manual = FALSE,
  vignettes = FALSE, RqmdRmd = FALSE, README = FALSE, NEWS = FALSE,
  ChangeLog = FALSE, checks = FALSE, targz = FALSE, untar = FALSE,
  binary = FALSE, type = "binary", script = FALSE, dir = ".",
  crandb = get("crandb", envir = .GlobalEnv),
  repos = getOption("repos")[1])

p_downarch(..., char = NULL, before = Sys.Date(), dir = ".",
  untar = FALSE, url = "https://cran.r-project.org/src/contrib/Archive")
```

```
targz_down(ptargz, dir = ".", untar = FALSE,
  url = "https://cran.r-project.org/src/contrib/Archive")
```

Arguments

...	any format recognized by cnsc, including list. A vector or packages or a named list of packages (with names being the keywords).
char	(name to) a character vector or a list. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
index	logical. Download the html index page of each package.
manual	logical. Download the pdf manual.
vignettes	logical. Download the html and pdf vignettes, if they exist.
RqmdRmd	logical. Download the R, Rmd and qmd source files of the vignettes, if they exist.
README	logical. Download the README file, if it exists.
NEWS	logical. Download the NEWS file, if it exists.
ChangeLog	logical. Download the ChangeLog file, if it exists.
checks	logical. Download the CRAN checks file.
targz	logical. Download the *.tar.gz source file.
untar	logical. Decompress the downloaded tar.gz file.
binary	logical. Download the *.tgz (Mac OSX) or *.zip (Windows) binary file, depending the type value.
type	character. Either "mac.binary", "mac.binary.el-capitan", or "win.binary". The default, "binary", automatically detects the local OS and the variants between R-3.6.3, R-4.0.0 or (for Windows) gcc8. See the type section of utils::install.packages.
script	logical. Create a mini-script and save it in a *.R file.
dir	character. The directory in which the files are saved. Default value "." is the current directory.
crandb	data.frame crandb. The data.frame of CRAN packages.
repos	character. The address of your local CRAN.
before	character which can be converted to a Date, for instance "2017-05-14". Extract from CRAN archive the package(s) available before this date. Can be synchronized with the release dates of base-R versions listed at: https://CRAN.R-project.org/src/contrib/ and https://CRAN.R-project.org/package=rversions/readme/README.html
url	character. The url address of CRAN archive html file.
ptargz	character. A vector of package(s) with their version number and tar.gz extension stored in CRAN archive. These packages can be identified with l_targz.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

## Download the documentation in the "dirpkgs" directory. Flat representation.
dir <- file.path(tempdir(), "dirpkgs")
p_down(RWsearch, pdfsearch, sos, dir = dir, repos = "https://cloud.r-project.org")
list.files(dir, recursive = TRUE, full.names = TRUE)

## Download the documentation in subdirectories named after the keywords.
dir   <- file.path(tempdir(), "dirpkgslist")
(lst  <- s_crandb_list(thermodynamic, "chemical reaction"))
(lst2 <- lapply(lst, function(x) x[1:2]))
system.time(
  p_down(lst2, dir = dir, repos = "https://cloud.r-project.org")
)
list.files(dir, recursive = TRUE, full.names = TRUE)

## Download tar.gz files stored in CRAN archive.
dir <- file.path(tempdir(), "targzip")
p_downarch(fitur, zmatrix, NotAPkg, before = "2017-05-14", dir = dir)
targz_down("SVN_1.0.tar.gz", dir = dir, untar = TRUE)
list.files(dir, recursive = TRUE, full.names = TRUE)
```

---

p_graph *Network and Graphs of Package Dependencies*

Description

p_graphF calculates the (recursive/reverse) dependencies of a (vector of) package(s) and displays in the default browser a standard graph (F/Force in the networkD3 terminology) of the package dependencies. It combines the p_network and n_graphF functions.

p_graphF calculates the (recursive/reverse) dependencies of a (vector of) package(s) and displays in the default browser a Sankey graph (in the networkD3 terminology) of the package dependencies. It combines the p_network and n_graphS functions.

p_network returns the (recursive/reverse) dependencies of a (vector of) package(s) as a network of nodes and links.

n_graphF taks as input a network of package nodes and links and displays them in the default browser as a standard graph (F/Force in the networkD3 terminology) representing the package dependencies.

n_graphF taks as input a network of package nodes and links and displays them in the default browser as a Sankey graph (in the networkD3 terminology) representing the package dependencies.

Remember that the option exclpkgs = ... whose default value TRUE is equivalent to exclpkgs = c("graphics", "grDevices", "methods", "stats", "tools", "utils"), can substantially modify the aspect of the graph, especially for reverse = FALSE.

Usage

```
p_graphF(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, exclpkgs = TRUE, group = 2, fontFamily = "serif",
  fontSize = 11, linkDistance = 50, charge = -100,
  crandb = get("crandb", envir = .GlobalEnv))

p_graphS(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, exclpkgs = TRUE, group = 2, fontFamily = "serif",
  fontSize = 14, nodeWidth = 30, nodePadding = 10,
  crandb = get("crandb", envir = .GlobalEnv))

p_network(..., char = NULL, which = "DIL", recursive = TRUE,
  reverse = FALSE, exclpkgs = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))

n_graphF(netw, group = 2, fontFamily = "serif", fontSize = 11,
  linkDistance = 50, charge = -100)

n_graphS(netw, group = 2, fontFamily = "serif", fontSize = 14,
  nodeWidth = 30, nodePadding = 10)
```

Arguments

`...`	any format recognized by cnsc. Lists are accepted for p_graphF and p_graphS (and will result in multiple html pages) but not in p_network. A vector or a list of package(s) listed in crandb or in installed.packages().
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
which	character vector. A sub-vector of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"). The short forms "D", "I", "L", "S", "N", "DL", "DI", "DIL", "DILS", "DILN", "DILSN", "SN" are accepted. "N" is for "Enhances" as the single letter "E" is used by R as a shortcut to EXPR, a reserved word.
recursive	logical. Search for (reverse) dependencies of (reverse) dependencies.
reverse	logical. Search for reverse dependencies.
exclpkgs	logical or character vector. TRUE excludes from the network of nodes and links the dependencies c("graphics", "grDevices", "methods", "stats", "tools", "utils"). FALSE includes them. You can provide your own vector of packages to exclude them from the network of nodes and links, for instance: exclpkgs = c("ggplot2", list.files(.Library)).
group	integer, currently 1, 2 or 3. The suffix of the "NGroup" column in *netw*. Define a scheme for colouring the nodes.
fontFamily	character. Either "serif" or "sans-serif".
fontSize	integer. The size of the font.
linkDistance	integer. The minimal distance of a link between two nodes.

charge	integer. A repulsive value between two nodes.
crandb	data.frame crandb. Also accepted is NULL which will search in the local installed.packages(). This later form allows (private) packages that are not listed in crandb.
nodeWidth	integer. The width of the rectangular nodes in the Sankey graph.
nodePadding	integer. The vertical space between two nodes in the same column of a Sankey graph.
netw	a list of class "pkgsnetwork" produced by p_network that describes the dependencies of one or several packages with nodes and links (a network).

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

lst  <- list("RWsearch", "visNetwork") ; lst
netw <- p_network(actuar, fitdistrplus, reverse = TRUE) ; netw

if (interactive()) {
p_graphF(lst)
p_graphS(RWsearch, visNetwork)

n_graphF(netw)
n_graphS(netw)
}
```

---

p_html *HTML Help Page, PDF Manual and Vignettes*

Description

p_page opens the default browser, connects to your local CRAN and displays the home page of the package(s). An internet connexion is required.

p_html and p_html2 open the default browser and display the html help page of the package, if it is installed. On Windows, p_html returns a local server address *http://127.0.0.1:*.html* and subfunctions listed in the page can be explored whereas p_html2 returns a file address *file:///C:/*.html* with no links to the subfunctions.

p_man and its alias p_htmlweb open the default browser and display the html help pages stored by the R-project at *https://search.r-project.org*. An internet connexion is required. A message is returned if the package does not exist in CRAN. In such case, use the function s_man for a deeper exploration.

p_pdf displays in a pdf reader the pdf manual of the package, or generates it on the fly in the current directory if the package is installed. Miktex or Texlive is required. This is a very fast function if the files already exist (and overwrite= FALSE) and a (relatively) slow function if the files needs to be generated, usually much slower than:

p_pdfweb downloads from you local CRAN the pdf manual of the package, saves it in the current directory and opens it in the pdf application of your browser. An internet connexion is required.

p_vig is a wrapper of utils::browseVignettes. It opens the default browser and displays a list of the vignettes related to a package, if they exist.

p_vig_all wraps utils::browseVignettes(NULL). It opens the default browser and displays all vignettes available in the computer. This can be a very large html file.

Use p_archive or p_archive to display in the browser or in the console the package archives. An internet connexion is required.

Usage

```
p_page(..., char = NULL, repos = getOption("repos")[1])

p_html(..., char = NULL)

p_html2(..., char = NULL)

p_htmlweb(..., char = NULL)

p_man(..., char = NULL)

p_pdf(..., char = NULL, overwrite = FALSE, dir = ".")

p_pdfweb(..., char = NULL, repos = getOption("repos")[1])

p_vig(..., char = NULL)

p_vig_all()
```

Arguments

...	any format recognized by cnsc, except list. A vector of packages.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
repos	character. The address of your local CRAN.
overwrite	logical. Overwrite already existing file (and use LaTeX intensively).
dir	character. The directory in which the files are read or written. Default value "." is the current directory.

Examples

```
if (interactive()) {
p_page(RWsearch, sos, repos = "https://cloud.r-project.org")
p_html(RWsearch, sos)
p_man(RWsearch)
p_pdfweb(sos, repos = "https://cloud.r-project.org")

## Try
```

```
p_pdf(sos, dir = file.path(tempdir(), "ppdf"))
p_vig(RWsearch)
}
```

---

p_inst *Immediate or delayed package installation*

Description

p_inst is a wrapper around install.packages. It tries hard to select the most appropriate lib and stops in case of conflict. Use install.packages if any additional argument is needed.

p_inst_batsh writes one executable file .sh (Linux, macOS) or two files .bat, .R (Windows) that allow a delayed installation of old packages in a R –vanilla environment. This is useful if the default configuration of R defined in R/etc/Rprofile.site launchs additional packages with compiled code, since these packages cannot be updated without modifying the R/etc/Rprofile.site file.

p_oldpkgs is a wrapper around old.packages.

For the three above functions, if crandb loaded in .GlobalEnv, then the packages are sorted by their increasing number of dependencies.

... allows a non-standard evaluation of unquoted packages separated by commas.

Usage

```
p_inst(..., char = NULL, lib = NULL, repos = getOption("repos")[1],
  contriburl = NULL, dependencies = NA, type = getOption("pkgType"))

p_oldpkgs(type = "both", lib.loc = .libPaths()[1],
  repos = getOption("repos")[1])

p_inst_batsh(..., char = NULL, type = "binary", verbose = TRUE,
  dir = ".", lib = .libPaths()[1], repos = getOption("repos")[1])
```

Arguments

...	any format recognized by cnsc, excluding list. A vector of packages.
char	(name to) a character vector or a list. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
lib	character. The directory where to install the packages, usually one of the directories listed by .libPaths. If NULL, select automatically the most relevant directory.
repos	character. The address of your local CRAN.
contriburl	character. The address of your private repository.
dependencies	logical. FALSE skips the installation of dependencies. NA installs c("Depends", "Imports", "LinkingTo") dependencies.
type	character. Either "source", "both", "binary" (or its variants "mac.binary", "mac.binary.el-capitan", "win.binary").

lib.loc	character. The directory where to search for old packages.
verbose	Logical. Print information of the download process.
dir	character. The directory where to write the files .sh, .bat, .R.

Examples

```
tmpdir <- tempdir()
pkgs <- p_oldpkgs(type = "source", repos = "https://cloud.r-project.org")
pkgs <- c("brew","fs","XML","RWsearch")
p_inst_batsh(pkgs, type = "source", dir = tmpdir, repos = "https://cloud.r-project.org")
if (.Platform$OS.type == "windows") {
    unlink(file.path(tmpdir, "installrpkgs.R"))
    unlink(file.path(tmpdir, "installrpkgs.bat"))
} else unlink(file.path(tmpdir, "installrpkgs.sh"))
```

---

p_inun *List of Installed, Uninstalled and Non-Existing Packages*

Description

p_incrandb returns TRUE if all packages are listed in crandb and a vector of FALSE with the names of the packages not listed in crandb.

p_inun returns a list of packages installed or not installed in the computer.

p_inun_crandb checks if the packages exist or do not exist in crandb (see CRAN archives, Bioconductor, Github, your own packages).

The missing packages available on CRAN can be downloaded with p_down0, downloaded and checked (by R CMD check) with xfun::rev_check or installed with install.packages. The packages removed from CRAN but available in CRAN archive can be downloaded with p_downarch.

Usage

```
p_incrandb(..., char = NULL, crandb = get("crandb", envir = .GlobalEnv))

p_inun(..., char = NULL)

p_inun_crandb(..., char = NULL, crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

...	any format recognized by cnsc, including list. A vector or a list of packages.
char	(name to) a character vector or a list. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
crandb	data.frame crandb.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

## Check if packages are installed or not, and exist or not in crandb
p_incrandb(RWsearch, NotAPkg1, pacman, NotAPkg2, sos)
p_inun(RWsearch, NotAPkg1, pacman, NotAPkg2, sos)
p_inun_crandb(RWsearch, NotAPkg1, pacman, NotAPkg2, sos)
```

---

p_table2pdf *Package Information in Console and PDF Files*

Description

p_table returns a subset of crandb for the given packages and the selected columns, by default the Package name, the Title and the Description.

p_table2 has a preset value to 2 columns: "Package", "Title" and prints the results in the console with a left alignment.

p_table5 has a preset value to 5 columns: "Package", "Title", "Description", "Author", "Maintainer".

p_table7 has a preset value to 7 columns: "Package", "Version", "Published", "Title", "Description", "Author", "Maintainer".

table_pdf prints the results of p_table, p_table5 or p_table7 in pdf file(s). Miktex or Texlive is required.

p_table2pdf, p_table3pdf, p_table5pdf, p_table7pdf combine the above functions.

Usage

```
p_table(..., char = NULL, columns = c("Package", "Title", "Description"),
  crandb = get("crandb", envir = .GlobalEnv))

p_table2(..., char = NULL, crandb = get("crandb", envir = .GlobalEnv))

p_table5(..., char = NULL, crandb = get("crandb", envir = .GlobalEnv))

p_table7(..., char = NULL, crandb = get("crandb", envir = .GlobalEnv))

table_pdf(x, filename = "SelectedPkgs.tex", dir = ".",
  texops = "a4paper,landscape,10pt", pdf = TRUE, cleantex = TRUE,
  openpdf = TRUE, verbose = TRUE)

p_table2pdf(..., char = NULL, filename = "Selectedpkgs", dir = ".",
  texops = "a4paper,landscape,10pt", pdf = TRUE, cleantex = TRUE,
```

```
  openpdf = TRUE, verbose = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))

p_table3pdf(..., char = NULL, filename = "Selectedpkgs", dir = ".",
  texops = "a4paper,landscape,10pt", pdf = TRUE, cleantex = TRUE,
  openpdf = TRUE, verbose = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))

p_table5pdf(..., char = NULL, filename = "Selectedpkgs", dir = ".",
  texops = "a4paper,landscape,10pt", pdf = TRUE, cleantex = TRUE,
  openpdf = TRUE, verbose = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))

p_table7pdf(..., char = NULL, filename = "Selectedpkgs", dir = ".",
  texops = "a4paper,landscape,10pt", pdf = TRUE, cleantex = TRUE,
  openpdf = TRUE, verbose = TRUE, crandb = get("crandb", envir =
  .GlobalEnv))
```

Arguments

`...`	any format recognized by cnsc, including list. A vector or a list of packages.
`char`	(name to) a character vector. Use this argument if `...` fails or if you call the function from another function. If used, argument `...` is ignored.
`columns`	character vector. A sub-vector of `colnames(crandb)`. The short form "P", "T", "D", "PT", "PD", "TD", "PTD", "A", "M", "AM" describing the Package name, Title, Description, Author, Maintainer or a combination of them is accepted.
`crandb`	data.frame `crandb`. The data.frame of CRAN packages.
`x`	(list of) data.frame produced by `p_table` (with 3 columns), `p_table5` (5 columns) or `p_table7` (7 columns). If `x` is a list, the names of the list will be appended to `filename`.
`filename`	character. The file name (with or without extension).
`dir`	character. The directory in which the files are read or written. Default value `"."` is the current directory.
`texops`	character vector. Options passed to instruction `documentclass` in *.tex file.
`pdf`	logical. `FALSE` generates the *.tex file. `TRUE` generates both the *.tex and *.pdf files.
`cleantex`	logical. Remove the `.tex` file(s) (only if `pdf = TRUE`).
`openpdf`	logical. Open the generated *.pdf file(s) in a pdf viewer (only if `pdf = TRUE`).
`verbose`	logical. Print the path(s) to the generated file(s).

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
```

```
## Use a large console (useful for p_table2())
p_table2(pacman, pdfsearch, sos)
(lst <- s_crandb_list("thermodynamic", "chemical reaction", "distillation"))
p_table2(lst)

## Print the tables as pdf files and open them in a pdf viewer.
if (interactive()) {
dir <- file.path(tempdir(), "ptablepdf")
p_table5pdf(pacman, pdfsearch, sos, filename = "table5", dir = dir)
p_table7pdf(lst, filename = "table7", dir = dir, cleantex = FALSE)
}
```

---

p_text2pdf *Download Package Documentation in Text Files*

Description

p_text extracts from CRAN the most relevant information related to one or several packages and print them in a text file which can be tailored to various formats: *.txt, *.md, *.tex for further treatment.

p_text2md has preset values for markdown files.

p_text2tex has preset values for latex files.

p_text2pdf has preset values for pdf files.

Usage

```
p_text(..., char = NULL, filename = "txtpkgs.txt", dir = ".",
  beforetext = "", f_maintext = funmaintext, sep1 = "== ",
  sep2 = " ==", eol = "\n", README = TRUE, NEWS = TRUE,
  ChangeLog = TRUE, vignettes = TRUE, RqmdRmd = FALSE, DOI = FALSE,
  aftertext = "", editor = FALSE, pager = FALSE, verbose = TRUE,
  crandb = get("crandb", envir = .GlobalEnv),
  repos = getOption("repos")[1])

p_text2md(..., char = NULL, filename = "mdpkgs.md", dir = ".",
  beforetext = funheadermd(), f_maintext = funmaintext, sep1 = "# ",
  sep2 = "  ", eol = "  \n", README = TRUE, NEWS = TRUE,
  ChangeLog = TRUE, vignettes = TRUE, RqmdRmd = FALSE, DOI = FALSE,
  aftertext = "", editor = FALSE, pager = FALSE, verbose = TRUE,
  crandb = get("crandb", envir = .GlobalEnv),
  repos = getOption("repos")[1])

p_text2tex(..., char = NULL, filename = "texpkgs.tex", dir = ".",
  beforetext = funheadertex(), f_maintext = funmaintex,
  sep1 = "\\section{", sep2 = "}", eol = " \\\\\n",
```

```
  README = TRUE, NEWS = TRUE, ChangeLog = TRUE, vignettes = TRUE,
  RqmdRmd = FALSE, DOI = FALSE, aftertext = funfootertex(),
  editor = FALSE, pager = FALSE, verbose = TRUE, crandb = get("crandb",
  envir = .GlobalEnv), repos = getOption("repos")[1])

p_text2pdf(..., char = NULL, filename = "pdfpkgs.pdf", dir = ".",
  beforetext = funheadertex(), f_maintext = funmaintex,
  sep1 = "\\section{", sep2 = "}", eol = " \\\\\n",
  README = TRUE, NEWS = TRUE, ChangeLog = TRUE, vignettes = TRUE,
  RqmdRmd = FALSE, DOI = FALSE, aftertext = funfootertex(),
  cleantex = TRUE, openpdf = FALSE, verbose = TRUE,
  crandb = get("crandb", envir = .GlobalEnv),
  repos = getOption("repos")[1])
```

Arguments

`...`	any format recognized by cnsc, including list. A vector or packages or a named list of packages (with names being the keywords).
`char`	(name to) a character vector or a list. Use this argument if `...` fails or if you call the function from another function. If used, argument `...` is ignored.
`filename`	character. The file name with extension. If `...` (or `...`) is a list, the names of the list will be appended to `filename`.
`dir`	character. The directory in which the files are read or written. Default value `"."` is the current directory.
`beforetext`	character. The text written at the beginning of the file.
`f_maintext`	function name. The function used to extract the main text from `crandb` (supplied with no parenthesis).
`sep1`	character. The symbols written just before each package name.
`sep2`	character. The symbols written just after each package name. If used with with mardkdown, add two blank characters at the end to force a new line.
`eol`	character. The end of line for the main text (but not for the header and the footer). "\n" for text, " \n" for rmarkdown, " \\ \n" for latex.
`README`	logical. Write the line related to the README page, if it exists.
`NEWS`	logical. Write the line related to the NEWS page, if it exists.
`ChangeLog`	logical. Write the line related to the ChangeLog page, if it exists.
`vignettes`	logical. Write the lines related to the vignette(s), if they exist.
`RqmdRmd`	logical. Add to the vignettes the source files in R, Rmd or qmd format, if they exist.
`DOI`	logical. Write the line related to the DOI link, if it exists.
`aftertext`	character. The text written at the end of the file.
`editor`	logical. Open the text file with `editor`.
`pager`	logical. Open the text file with `pager`.
`verbose`	logical. List the generated file(s).

crandb	data.frame crandb. The data.frame of CRAN packages.
repos	character. The address of your local CRAN.
cleantex	logical. Remove the .tex file(s).
openpdf	logical. Open the pdf files in the default pdf viewer.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

## Search in crandb
vec <- s_crandb(search, find, select = "PT") ; vec
lst <- s_crandb_list(thermodynamic, "chemical reaction") ; lst
lst2 <- lapply(lst, function(x) x[1:2]) ; lst2
dir <- file.path(tempdir(), "ptext")

## Generate a txt file

p_text(vec[1:5], filename = "SearchFind.txt", dir = dir,
       RqmdRmd = TRUE, DOI = TRUE, repos = "https://cloud.r-project.org")

## Generate 2 tex + 2 pdf files (10-20 seconds)
## Try the options cleantex = FALSE and openpdf = TRUE on lst
if (interactive()) {
p_text2pdf(lst2, dir = dir, cleantex = TRUE, openpdf = FALSE,
           RqmdRmd = TRUE, DOI = TRUE, repos = "https://cloud.r-project.org")
}
```

---

p_unload_all *Unload all non-base and non-recommended packages from the namespace*

Description

p_unload_all unloads in a safe order all packages, except the base and recommended packages, that are attached or loaded in the namespace, plus their respective DLLs.

It must be used before installing new versions of packages that are currently loaded in the namespace and require some forced unloading, as revealed by the first column of the data.frame produced by p_vers_deps. Warning: this function also removes RWsearch and its dependencies. It is therefore recommended to perform the installation with install.packages rather than p_inst.

This function is safe enough to reinstall locked packages but not safe enough for a further use as it does not unload the S3 and S4 classes nor the documentation which is detected as corrupted. It is therefore recommend to stop and restart R after the installation of the fresh packages.

Usage

```
p_unload_all(unload = FALSE, crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

unload	logical. FALSE prints a list with 3 items: the loaded packages, the packages in the namespace, the loaded DLLs. TRUE removes the packages and their DLLs, then prints the refreshed list.
crandb	data.frame crandb.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

p_unload_all()
## Then run p_unload_all(TRUE), but only for maintenance!
```

---

p_vers *Package Version and Number of Dependencies*

Description

The information displayed by p_vers depends on the availability of crandb and binarydb in .GlobalEnv.

If crandb is not loaded in .GlobalEnv, p_vers returns a data.frame with two columns: first column nsloaded (TRUE/FALSE) detects (with base::isNamespaceLoaded) if the package namespaces are loaded. Second column version is the version number of the installed packages.

If crandb is loaded in .GlobalEnv, three columns are added. Column crandb displays the version number of the source packages recorded in the crandb file. Column gcc displays the nedd for a compilation. Column compare compares this version number with the version installed on the computer. Possible values are: -2 for a package not installed on the computer (NA) but available in crandb ; -1 for an installed package older than the source package available in CRAN ; 0 for an installed package with the same version number than CRAN ; +1 for a more recent package than the one available in CRAN ; +2 for a package installed on the computer and not available in CRAN (NA) ; +3 for a package not installed on the computer (NA) and not available in CRAN (NA).

If binarydb is loaded in .GlobalEnv, two or three columns are added. Column binary displays the version number of the binary packages recorded in the binarydb = available.packages, type = "binary") matrix. Column difvb compares the installed version on the computer with this binary version and column difbc compares (if crandb is in .GlobalEnv) the binary version with the source package available in CRAN (which can differ for recently updated packages, a matter of 1 to 3 days). The numbering is identical to the one used for crandb.

If ndeps = TRUE, two more columns are added with the number of recursive dependencies per package. Column tdeps includes the base and recommended packages. Column ndeps does not count them. This option can take some time. Set it to FALSE if you need speed.

p_vers_deps calculates the same information but includes the recursive dependencies. Subsetting by "compare < 0" returns a shorter data.frame with the uninstalled (-2) and the outdated (-1) packages. Packages marked with nsloaded = TRUE must be detached and unloaded before any reinstallation. Using this instruction before running install.packages or p_inst is very useful as it detects packages that are locked and cannot be reinstalled. The order provided by p_vers_deps is the best one for the reinstallation of outdated packages.

Usage

```
p_vers(..., char = NULL, ndeps = TRUE, split = FALSE, reserved = "")

p_vers_deps(..., char = NULL, ndeps = TRUE, subset = "compare < 4",
  crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

...	any format recognized by cnsc. A (list of) vector of packages for p_vers. A vector of packages for p_vers_deps
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
ndeps	logical. Calculate the number of recursive dependencies. crandb in .GlobalEnv is required for this option.
split	logical. Split the output in a list of data.frame with packages allocated in at most 6 sections : library, reserved, nsloaded, cran, source, binary.
reserved	character. The packages in the reserved section.
subset	character. Subset the output data.frame on some columns. The default "compare < 4" does not subset. Usual values are "compare < 0" or "compare < 0 && nsloaded == TRUE"
crandb	data.frame crandb. The data.frame of CRAN packages.

Examples

```
pkgs <- cnsc(RWsearch, MASS, Matrix, NotAPkg, R)
p_vers(pkgs)

## Now with crandb and binarydb loaded in .GlobalEnv. In real life, use:
## crandb_down() ; binarydb_down()
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
binarydb_load(system.file("data", "zbinarydb.rda", package = "RWsearch"))

p_vers(pkgs)
p_vers(p_deps(pkgs))
p_vers_deps(pkgs)    # Dependencies can be visualized with p_graphF(pkgs)
p_vers(char=c(p_depsrec(RWsearch)$RWsearch, "RWsearch"), split = TRUE)
```

s_crandb *Search Packages by Keywords in data.frame crandb*

Description

The most important functions in this package along with p_down.

Search packages in data.frame crandb that contain one or several keywords in the columns "Package", "Title", "Description", "Author" or "Maintainer".

s_crandb returns a vector of the packages that contain the keywords.

s_crandb_list returns a list where each element of the list is one of the keywords.

s_crandb_PTD returns a list split by results in columns "Package", "Title" and "Description". Option mode = "and", "relax" is ignored.

s_crandb_AM returns a list split by results in columns "Author" and "Maintainer". Option mode = "and", "relax" is ignored.

Use p_table2 to print the results of s_crandb and s_crandb_list in the console. Use p_text to send the results in txt, md or pdf files. Use p_display to visualize the results in html pages in the browser.

Usage

```
s_crandb(..., char = NULL, select = "PTD", mode = "or",
  sensitive = FALSE, perl = FALSE, fixed = FALSE, agrep = FALSE,
  max.distance = 0.1, costs = NULL, crandb = get("crandb", envir =
  .GlobalEnv))

s_crandb_list(..., char = NULL, select = "PTD", mode = "or",
  sensitive = FALSE, perl = FALSE, fixed = FALSE, agrep = FALSE,
  max.distance = 0.1, costs = NULL, crandb = get("crandb", envir =
  .GlobalEnv))

s_crandb_PTD(..., char = NULL, mode = "or", sensitive = FALSE,
  perl = FALSE, fixed = FALSE, agrep = FALSE, max.distance = 0.1,
  costs = NULL, crandb = get("crandb", envir = .GlobalEnv))

s_crandb_AM(..., char = NULL, mode = "or", sensitive = FALSE,
  perl = FALSE, fixed = FALSE, agrep = FALSE, max.distance = 0.1,
  costs = NULL, crandb = get("crandb", envir = .GlobalEnv))
```

Arguments

...	any format recognized by cnsc, except list. One or several keywords.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.

select	character vector. A sub-vector of colnames(crandb). The short form "P", "T", "D", "PT", "PD", "TD", "PTD", "A", "M", "AM" describing the Package name, Title, Description, Author, Maintainer or a combination of them is accepted.
mode	character among "or", "and", "relax". The search mode. "relax" is for 3 words and more. It is an intermediate between "or" and "and" as it requires just 2 matching words: ("word1" AND "word2") OR ("word1" AND "word3") OR ("word1" AND "word3").
sensitive	logical. TRUE forces the search to be case sensitive.
perl	logical. Used only if fixed = FALSE. TRUE uses Perl-compatible regex. FALSE uses default regexps.
fixed	logical. TRUE matchs the keywords as is (and sensitive is forced to TRUE). FALSE allows grep or Perl regexps. See grep. Not used by agrep.
agrep	logical. For approximate matching, use agrep function rather than grep.
max.distance	integer or numeric. See agrep.
costs	NULL or list. See agrep.
crandb	data.frame crandb.

Examples

```
## In real life, download crandb from CRAN or load it from your directory
## with functions crandb_down() or crandb_load().
## In this example, we use a small file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))

## Search using standard or non-standard content
s_crandb(c("thermodynamic", "chemical reaction", "distillation"))
s_crandb_list(thermodynamic, "chemical reaction", distillation)

## Search using the various options
s_crandb("^f", select = "P")
s_crandb(pH, sensitive = TRUE)
s_crandb_PTD(pH, sensitive = TRUE)
s_crandb_PTD("C++", fixed = TRUE)
s_crandb(search, find, cran, web, select = "PD", mode = "and")
s_crandb(search, find, cran, web, select = "PD", mode = "relax")
s_crandb(search, find, cran, web, select = "PD", mode = "or")

## Search for some authors using the various options
s_crandb_AM(Kiener, Dutang, ORPHANED)

## Non-standard content can be unquoted words or objects in .GlobalEnv
## They are transformed into character or are evaluated
## Here, the searched keywords are "find" and "search".
OTHER <- "search"
(lst <- s_crandb_list(find, OTHER, select = "P", sensitive = TRUE))

## Display in the browser this list of packages
if (interactive()) {
p_display5(lst, dir = file.path(tempdir(), "scrandb"))
```

```
}
```

`s_crandb_tvdb` *Search For Recent Packages In crandb And In Task View*

Description

This is a function for task view maintenance.

Search packages in a subset of `crandb` within dates `from` and `to` that contain one or several keywords in the columns "Package", "Title", "Description", 'Author" or "Maintainer", then verify if these packages are already refereed in one of the task views stored in `tvdb`.

Usage

```
s_crandb_tvdb(..., char = NULL, tv = "Distributions", from = -10,
  to = Sys.Date(), select = "PTD", mode = "or", sensitive = FALSE,
  perl = FALSE, fixed = FALSE, agrep = FALSE, max.distance = 0.1,
  costs = NULL, crandb = get("crandb", envir = .GlobalEnv),
  tvdb = get("tvdb", envir = .GlobalEnv))
```

Arguments

`...`	any format recognized by `cnsc`, except list. One or several keywords.
`char`	(name to) a character vector. Use this argument if `...` fails or if you call the function from another function. If used, argument `...` is ignored.
`tv`	character. One task view among those listed in tvdb.
`from`	character representing a date earlier than date `to`. Or a negative integer representing the number of days preceeding the date `to`.
`to`	date. The upper date in the search.
`select`	character vector. A sub-vector of `colnames(crandb)`. The short form "P", "T", "D", "PT", "PD", "TD", "PTD", "A", "M", "AM" describing the Package name, Title, Description, Author, Maintainer or a combination of them is accepted.
`mode`	character among "or", "and", "relax". The search mode. "relax" is for 3 words and more. It is an intermediate between "or" and "and" as it requires just 2 matching words: `("word1" AND "word2") OR ("word1" AND "word3") OR ("word1" AND "word3")`.
`sensitive`	logical. `TRUE` forces the search to be case sensitive.
`perl`	logical. Used only if `fixed = FALSE`. `TRUE` uses Perl-compatible regex. `FALSE` uses default regexps.
`fixed`	logical. `TRUE` matchs the keywords as is (and `sensitive` is forced to `TRUE`). `FALSE` allows grep or Perl regexps. See `grep`. Not used by `agrep`.
`agrep`	logical. For approximate matching, use `agrep` function rather than grep.

max.distance	integer or numeric. See agrep.
costs	NULL or list. See agrep.
crandb	data.frame crandb.
tvdb	list. The list of the task views.

Value

A list with the following vectors:

- spkgs: The selected packages that contain the keyword(s).
- inTV: The packages that contain the keyword(s) already refereed in the task view.
- notinTV: The packages that contain the keyword(s) not (yet) refereed in the task view.
- inTV_in: Among the packages available in the task view, those installed in the computer.
- inTV_un: Among the packages available in the task view, those not installed in the computer.
- notinTV_in: Among the packages not refereed in the task view, those installed in the computer.
- notinTV_un: Among the packages not refereed in the task view, those not installed in the computer.

Examples

```
### TASK VIEW MAINTENANCE (tvdb + crandb)
## In real life, download crandb and tvdb from CRAN or load them from your directory
## with functions crandb_down(), crandb_load(), tvdb_down(), tvdb_load().
## In this example, we use small files.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
tvdb_load(system.file("data", "ztvdb.rda", package = "RWsearch"))

## List the task views
tvdb_vec()

## Search for the recent packages in crandb that contain the keyword
## and verify that the packages are already refereed in the task view.
(lst <- s_crandb_tvdb("Gumbel", tv = "Distributions", from = "2018-01-01"))

if (interactive()) {
p_display7(lst[c("inTV", "notinTV")], dir = file.path(tempdir(), "scrandbtvdb"))
}
```

---

s_hs *Search Packages and Functions in Installed Packages*

Description

s_hs is a wrapper of the well known function ?? and its parent function help.search. Visit the help page help.search for details on the various arguments.

Usage

```
s_hs(..., char = NULL, fields = c("alias", "concept", "title"), apropos,
  keyword, whatis, ignore.case = TRUE, package = NULL, agrep = NULL,
  use_UTF8 = FALSE)
```

Arguments

...	one single character string recognized by cnsc. One and only one pattern
char	(name to) a single character string. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
fields	See help.search.
apropos	See help.search.
keyword	See help.search.
whatis	See help.search.
ignore.case	See help.search.
package	See help.search.
agrep	See help.search.
use_UTF8	See help.search.

---

s_sos *Search Packages and Functions on R-Project Help pages and RDocumentation*

Description

s_sos searchs in all R documentation packages and functions that contain one or several keywords, open the default browser and display the results in two html pages, one for the package::functions and one for the package::vignettes. s_sos usually find more results than s_crandb as it goes deeper in the documentation, down to the function level. An internet connection is required to reach the website https://search.r-project.org. s_sos is a minimal wrapper of the function *sos::findFn*. Use this function if you need advanced search options.

s_man sends a query to the same search engine but the results is returned in a less edited format than s_sos. Use the function p_man if your search is about a package and not a function.

Usage

```
s_sos(..., char = NULL)

s_man(..., char = NULL)
```

Arguments

...	any format recognized by cnsc, except list. One or several keywords.
char	(name to) a character vector. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.

See Also

https://CRAN.R-project.org/package=sos (index and vignette).

Examples

```
## Search using standard or non-standard content
## and display the results in a browser.
if (interactive()) {
s_sos("chemical reaction")
(res <- s_sos(distillation))
tail(data.frame(res))
s_man("cran_incoming")
}
```

---

s_tvdb *Search Packages in Task Views*

Description

s_tvdb searchs if one or several package(s) are referred in some task views and lists these task views.

Usage

```
s_tvdb(..., char = NULL, tvdb = get("tvdb", envir = .GlobalEnv))
```

Arguments

...	any format recognized by cnsc, except list. The names of one or several task views.
char	(name to) a character vector or a list. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.
tvdb	list. The list of the task views.

Examples

```
## In real life, download tvdb from CRAN or load it from your directory
## with functions tvdb_down() or tvdb_load().
## In this example, we use a small file.
tvdb_load(system.file("data", "ztvdb.rda", package = "RWsearch"))
tvdb_dfr()
s_tvdb(actuar, FatTailsR, MASS, zoo, NotAPkg)
```

tvdb *Task Views (tvdb.rda)*

Description

tvdb_down downloads from CRAN the file "Views.rds", a file refreshed every day that describes the task views available in CRAN for this day, rearranges the list in an alphabetical order and gives names to the list names, then loads in .GlobalEnv this list (of class ctvlist) under the name tvdb and saves it with the filename tvdb.rda.

tvdb_load loads the file filename in .GlobalEnv under the name tvdb. Equivalent to load("tvdb.rda").

tvdb_vec displays the list of the task views. There are 36 task views in August 2018.

tvdb_dfr extracts from tvdb a data.frame *version, name, topic* of the task views.

tvdb_list extracts from tvdb the list of the task views and the referenced packages.

tvdb_pkgs displays the packages referenced by one or several task views.

Visit s_crandb_tvdb to conduct task view maintenance.

Usage

```
tvdb_down(dir = ".", repos = getOption("repos")[1])

tvdb_load(filename = "tvdb.rda")

tvdb_vec(tvdb = get("tvdb", envir = .GlobalEnv))

tvdb_dfr(tvdb = get("tvdb", envir = .GlobalEnv))

tvdb_list(tvdb = get("tvdb", envir = .GlobalEnv))

tvdb_pkgs(..., char = NULL, tvdb = get("tvdb", envir = .GlobalEnv))
```

Arguments

dir	character. The directory where "tvdb.rda" is saved. Default value "." is the current directory.
repos	character. The address of your local CRAN.
filename	character. The path to file "tvdb.rda". The default is to read in the current directory.
tvdb	list. The list of the task views.
...	any format recognized by cnsc, except list. The names of one or several task views.
char	(name to) a character vector or a list. Use this argument if ... fails or if you call the function from another function. If used, argument ... is ignored.

Examples

```
### DOWNLOAD AND VISUALIZE THE TASK VIEWS (tvdb)
## In real life, download tvdb from CRAN or load it from your directory
## with functions tvdb_down() or tvdb_load().
## In this example, we use a small file.
tvdb_load(system.file("data", "ztvdb.rda", package = "RWsearch"))
length(tvdb)

## List the task views
tvdb_vec()
tvdb_dfr()
tvdb_pkgs("Robust")
lengths(tvdb_list())

## Here, 'lst' is subsetted from the small crandb file.
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
funIN <- function (x, y) x[match(x, y, nomatch = 0)  > 0]
lst   <- lapply(tvdb_list()[1:2], funIN, crandb$Package) ; lst

if (interactive()) p_display7(lst[[1]], dir = file.path(tempdir(), "ptvdbdown"))
```

---

zcrandb *File zcrandb.rda: A Subset of crandb Dataset*

Description

The file *zcrandb.rda* contains a data.frame named `crandb` of dimension 94 x 69. It is is a subset of 94 packages of the large `crandb` data.frame usually downloaded from CRAN by the function `crandb_down`. The use of `zcrandb.rda` avoids inappropriate connections to CRAN and increases the speed of the examples.

Examples

```
crandb_load(system.file("data", "zcrandb.rda", package = "RWsearch"))
```

---

ztvdb *File ztvdb.rda: A Subset of tvdb Dataset*

Description

The file *ztvdb.rda* contains a list of 4 task views named `tvdb`. It is a subset of the large file *tvdb.rda* that contain 45 task views usually downloaded from CRAN by the function `tvdb_down`. The use of *ztvdb.rda* avoids inappropriate connections to CRAN and increases the speed of the examples.

Examples

```
tvdb_load(system.file("data", "ztvdb.rda", package = "RWsearch"))
```

Index