

Package ‘sos’

July 23, 2025

Title Search Contributed R Packages, Sort by Package

Version 2.1-8

Date 2024-06-28

Author Spencer Graves [cre, aut, cph],
Sundar Dorai-Raj [aut], and Romain Francois [ctb]

Maintainer Spencer Graves <spencer.graves@prodsyse.com>

Description Search contributed R packages, sort by package.

License GPL (>= 2)

Depends R (>= 3.5.0), brew

Suggests RODBC, WriteXLS, testthat, knitr, rmarkdown

Language en-us

URL <https://github.com/sbgraves237/sos>

BugReports <https://github.com/sbgraves237/sos/issues>

VignetteBuilder knitr, rmarkdown

NeedsCompilation no

Repository CRAN

Date/Publication 2024-06-29 18:40:02 UTC

Contents

back2ForwardSlash	2
CRAN	3
Extract.findFn	5
findFn	6
grepFn	9
hits	10
installPackages	11
packageSum	13
PackageSum2	15
PackageSummary	17

print.findFn	18
print.packageSum	20
sortFindFn	21
summary.findFn	22
unionFindFn	24
writeFindFn2xls	25
Index	27

back2ForwardSlash	<i>Replace backslash with forward slash in a character string</i>
-------------------	---

Description

scan a character string with backslash as the quote character and return it with backslashes replaced by forward slash.

NOTE: 'c:\User' cannot be assigned to a character variable, because '\U' must be followed by a hexadecimal number, and 's' is not a legal hexadecimal digit. Therefore, we read the character string of interest using scan rather than assigning it to a function argument.

Usage

```
back2ForwardSlash(nmax=1, what=character(),  
  sep='\n', ...)
```

Arguments

nmax, what, sep, ...
arguments passed to [scan](#)

Details

It's not easy to turn a back slash into a forward slash, because R interprets the back slash as an escape character. back2ForwardSlash tells R to read the next nmax lines, replacing '\ ' with '/ '.

Value

character vector with backslashes replaced by forward slashes.

Author(s)

Spencer Graves with help from Richard Cotton and Garrett See.

See Also

[scan gsub Quotes](#)

Examples

```
(x <- back2ForwardSlash())
#c:\users\

#NOTE: The "#" in this example is not needed.
# It is included here to suppress a spurious warning
# in the automated testing of the package via "R CMD check".

all.equal(x, '#c:/users/')

(x2. <- back2ForwardSlash(2))
#c:\u\ a b\n o
#d:\pqr\

all.equal(x2., c('#c:/u/a b/n o', '#d:/pqr/'))
```

CRAN

Test if running as CRAN

Description

This function allows package developers to run tests themselves that should not run on CRAN or with "R CMD check --as-cran" because of compute time constraints with CRAN tests.

Usage

```
CRAN(CRAN_pattern, n_R_CHECK4CRAN)
```

Arguments

CRAN_pattern	a regular expressions to apply to the names of <code>Sys.getenv()</code> to identify possible CRAN parameters. Defaults to <code>Sys.getenv('_CRAN_pattern_')</code> if available and <code>'^_R_'</code> if not.
n_R_CHECK4CRAN	Assume this is CRAN if at least n_R_CHECK4CRAN elements of <code>Sys.getenv()</code> have names matching x. Defaults to <code>Sys.getenv('_n_R_CHECK4CRAN_')</code> if available and 5 if not.

Details

The "Writing R Extensions" manual says that "R CMD check" can be customized "by setting environment variables `_R_CHECK_*_:`, as described in" the Tools section of the "R Internals" manual.

'R CMD check' was tested with R 3.0.1 under Fedora 18 Linux and with Rtools 3.0 from April 16, 2013 under Windows 7. With the '--as-cran' option, 7 matches were found; without it, only 3 were found. These numbers were unaffected by the presence or absence of the '--timings' parameter. On this basis, the default value of `n_R_CHECK4CRAN` was set at 5.

1. `x. <- Sys.getenv()`
2. Fix `CRAN_pattern` and `n_R_CHECK4CRAN` if missing.
3. Let `i` be the indices of `x.` whose names match all the patterns in the vector `x.`
4. Assume this is CRAN if `length(i) >= n_R_CHECK4CRAN`.

Value

a logical scalar with attributes '`Sys.getenv`' containing the results of `Sys.getenv()` and '`matches`' containing `i` per step 3 above.

Author(s)

Spencer Graves (copied from the `fda` package)

See Also

[Sys.getenv skip_on_cran](#), which uses ["the NOT_CRAN env var set by devtools and friends"](<https://testthat.r-lib.org/reference/skip.html>). This CRAN function does NOT require a user to set any environment variable.

Examples

```
cran <- CRAN()
str(cran)
gete <- attr(cran, 'Sys.getenv')
(ngete <- names(gete))

iget <- grep('^_', names(gete))
gete[iget]

#\dontrun is sometimes run on CRAN. See
#https://github.com/ThinkR-open/prepare-for-cran
#accessed 2021-06-14
if (interactive()) {
  if(CRAN()){
    stop('CRAN')
  } else {
    stop('NOT CRAN')
  }
}
```

Extract.findFn	<i>Subset a findFn object</i>
----------------	-------------------------------

Description

Extract rows from a `findFn` object

Usage

```
## S3 method for class 'findFn'
x[i, j,
  drop =
    if (missing(i)) TRUE else length(j) == 1]
```

Arguments

<code>x</code>	An object of class <code>findFn</code>
<code>i</code>	a valid object to select rows of <code>x</code> , e.g., a vector of all positive integers or all negative integers between 1 and <code>nrow(x)</code> or a logical vector of length <code>nrow(x)</code> .
<code>j</code>	If not missing, the extraction function returns an object of class <code>data.frame</code> rather than <code>findFn</code> .
<code>drop</code>	logical: if <code>FALSE</code> and <code>j</code> selects only one column, return that column as a vector; else return a <code>data.frame</code> if <code>j</code> is present or a <code>findFn</code> object otherwise.

Details

1. if(`missing(j)`) extract the subset with the `PackageSummary` attribute recomputed on the subset.
2. else `return(Extract.data.frame(x, i, j, drop))`

Value

If `j` is missing, return an object of class `c('findFn', 'data.frame')` else return whatever is returned by `Extract.data.frame`.

Author(s)

Spencer Graves

See Also

`findFn`, `data.frame`

Examples

```
z <- try(findFn("spline", maxPages = 2))

if(!inherits(z, "try-error")){
  # one row
  z1 <- z[1,]

  # one column
  z.2 <- z[, 2]
  #
  z.2a <- z[2]

  all.equal(z.2, z.2a)

  # data.frame with one column
  z.2d <- z[, 2, drop=FALSE]
}
```

findFn

Search Help Pages

Description

Returns a data.frame from [RSiteSearch](#)(string, "function") which can be sorted and subsetted by user specifications and viewed in an HTML table. The default sort puts first packages with the most matches (Count), with ties broken using the sum of the match scores for all the hits in that package (TotalScore), etc.

Usage

```
findFn(string, maxPages = 100, sortby = NULL,
       verbose = 1, ...)
```

Arguments

string	A character string. See RSiteSearch .
maxPages	The maximum number of pages to download assuming 20 links per page.
sortby	a character vector specifying how the data.frame returned should be sorted. Default = c('Count', 'MaxScore', 'TotalScore', 'Package', 'Score', 'Function') to sort descending on numerics and ascending on alphanumerics. Specifying sortby = c('c', 't', 'm') is equivalent to c('Count', 'TotalScore', 'MaxScore', 'Package', 'Score', 'Function').
verbose	an integer: if 0, no output is printed to the console. The default 1 displays an initial line with the number of pages to be retrieved and the number of matches obtained; if the number of matches to be downloaded is less, this also is displayed on the initial line. This is followed by a second line counting the pages downloaded. If greater than 1, additional information is provided on the download process.

... ignored

Details

findFn searches the help pages of packages covered by the RSiteSearch archives. To restrict the search to only packages installed locally, use [help.search](#).

1. Access the [RSiteSearch](#) engine with string, restricting to "functions", storing Score, Package, Function, Date, Description, and Link in a data.frame.
2. Compute Count, MaxScore and TotalScore for each Package accessed. Combine them in a matrix PackageSummary.
3. Sort PackageSummary in the order defined by the occurrence of c('Count', 'MaxScore', 'TotalScore', 'Package') in sortby.
4. Merge PackageSummary with the data.frame of search matches.
5. Sort the combined data.frame as defined by sort..
6. Make the result have class c("findFn", "data.frame") and add attributes matches, PackageSummary, string, and call.
7. Done.

Value

an object of class c('findFn', 'data.frame') with columns and attributes as follows:

Columns	• Count Total number of matches downloaded in this package • MaxScore maximum of the Score over all help pages selected within each Package. See Score below or the Namazu website (link below) for more information on how the score is determined. • TotalScore sum of the Score over all help pages selected within each Package. See Score below or the Namazu website (link below) for more information on how the score is determined. • Package Name of the package containing a help page meeting the search criteria • Function Name of the help page found that meets the indicated search criterion. • Date Date of the help page • Score Score returned by RSiteSearch, discussed in the Namazu website (link below). • Description Title of the help page • Link Universal Resource Locator (URL) for the help page
Attributes	• matches an integer = total number of matches found by the search. This typically will exceed the number of rows found, because the search algorithm sometimes finds things that are not help pages for packages. • PackageSummary a data.frame with one row for each package and columns Package, Count, MaxScore, TotalScore, and Date, sorted as in the sort. argument. • string the string argument in the call. • call the matched call

Author(s)

Spencer Graves, Sundar Dorai-Raj, Romain Francois. Duncan Murdoch suggested the "???" alias for findFn and contributed the code for it.

Special thanks to Gennadiy Starostin, Vienna University of Economics and Business (Wirtschaftsuniversitaet Wien), who in early 2021 took over maintenance of the [RSiteSearch](#) data base, updated its structure, and rewrote findFn to match.

Special thanks to Jonathan Baron, and Andy Liaw. Baron maintained the RSiteSearch data base for many years. Liaw and Baron created the RSiteSearch function in the utils package. Thanks also to Patrice Kiener of 'InModelia' in Paris, France, who helped me fix some syntax problems stemming from changes in how an itemized list is described in a *.Rd file.

References

<http://www.namazu.org/doc/tips.html.en#weight> - reference on determining Score

See Also

[help.search](#) to search only installed packages. [RSiteSearch](#), [download.file](#) findFn searches only "Target: Functions" from that site, ignoring the R-help archives.

For alternative R search capabilities, see:

* ["Searching R Packages" on Wikiversity](#)

* [Julia Silge, John C. Nash, and Spencer Graves \(2018\) Navigating the R Package Universe, The R Journal, 10\(2\) 558-563.](#)

* <https://search.r-project.org> for a list of alternative R search capabilities, each of which may be best for different types of inquiries.

* [findFunction](#) for a completely different function with a similar name.

Examples

```
# Skip these tests on CRAN,
# because they take more than 5 seconds
if(!CRAN()){

  z <- try(findFn("spline", maxPages = 2))
  # alternative
  zq <- try(???spline(2))

  # Confirm z == zq except for 'call'
  attr(z, 'call') <- NULL
  attr(zq, 'call') <- NULL

  if(!(inherits(z, "try-error") ||
    inherits(zq, "try-error"))){

    all.equal(z, zq)
```



```

# To search for 2 terms, not necessarily together:
RSS <- try(findFn('RSiteSearch function', 1))
matches(RSS)

# To search for an exact string, use braces:
RSS. <- try(findFn('{RSiteSearch function}', 1))
matches(RSS.) # list(nrow = 0, matches = 0)

# example in which resulting page has some unicode characters
Lambert <- try(findFn("Lambert"))
Lambert

# Example that "found 2 link(s) without dates" on 2021-06-26
webScr <- try(findFn('web scraping'))

# Example that "found 0 matches" on 2021-09-06
try(findFn('{open history map}'))
}
}

```

grepFn

Match pattern in a column of a matrix or data.frame

Description

Search for pattern in a column of a matrix or data.frame using grep. If value = TRUE (the default), return the selected subset of x.

Usage

```

grepFn(pattern, x, column='Function', ignore.case=FALSE,
        perl=FALSE, value=TRUE, fixed=FALSE,
        useBytes=FALSE, invert=FALSE)

```

Arguments

x	a matrix or data.frame containing a column named column.
pattern, ignore.case, perl, fixed, useBytes, invert	as for grep
column	character string giving the column of x in which to search for pattern.
value	logical: If TRUE, return the selected subset of x. If FALSE, return the row numbers returned by grep.

Details

1. g <- grep(pattern, x[, column])
2. if(value)return(x[g,]) else return(g)

Value

If(value) return an object of the same class as x containing those rows of x with x[, column] matching pattern.

Else, return an integer vector identifying the rows of x with x[, column] matching pattern.

Author(s)

Spencer Graves, Sundar Dorai-Raj

See Also

[findFn](#) [grep](#)

Examples

```
z <- cbind(a=1:2, Function=c('s', 'spline'))
z. <- grepFn("spline", z)
```

```
all.equal(z., z[2,,drop=FALSE])
```

hits

matches attribute of a findFn object

Description

Returns the matches attribute of a [findFn](#) object. For the output of findFn, this is the number of matches for the search term. For a findFn object returned by [unionFindFn](#) or [intersectFindFn](#), this is a numeric vector if the matches attributes of the arguments to unionFindFn or intersectFindFn.

Usage

```
matches(x)
hits(x)
```

Arguments

x object of class findFn.

Details

```
nrow(x) attr(x, 'matches')
```

Value

a list with components nrow and matches

Author(s)

Spencer Graves

See Also[findFn](#) [unionFindFn](#) [intersectFindFn](#)**Examples**

```
des1 <- try(findFn('differential equations', 1))

if(inherits(des1, 'try-error')){
  des1. <- matches(des1)
  des. <- list(nrow=nrow(des1), matches=attr(des1, 'matches'))

  all.equal(des1., des.)
}
```

installPackages

*install packages with minimum count***Description**

Ensure that the most important packages in `x` are installed. "Importance" here is defined in the description of the `minCount` argument below.

Usage

```
installPackages(x, minCount, ...)
## S3 method for class 'findFn'
installPackages(x, minCount, ...)
## S3 method for class 'packageSum'
installPackages(x, minCount,
               repos = getOption("repos"), ...)
```

Arguments

<code>x</code>	either a character vector to be passed to install.packages or a <code>findFn</code> or a <code>packageSum</code> object
<code>minCount</code>	Controls how many of the packages identified in <code>x</code> to pass to install.packages . If <code>x</code> is a <code>findFn</code> or <code>packageSum</code> object, install every <code>x[, 'Package']</code> with <code>x[, 'Count'] >= minCount</code> . By default, <code>minCount = sqrt(x[1, 'Count'])</code> .
<code>repos</code>	argument passed to install.packages
<code>...</code>	optional arguments passed to install.packages

Details

Functions `PackageSum2` and `packageSum` obtain some of the information displayed from installed packages. To get more information in those summaries, run `installPackages` on a `findFn` or `packageSum` object to install more of the packages found.

Value

none

Author(s)

Spencer Graves

See Also

[install.packages PackageSum2](#)

Examples

```
##
## 1. findFn object
##
spl <- try(findFn("spline", maxPages = 2))

if(!inherits(spl, "try-error")){
  # check the code but do not install anything:
  installPackages(spl, minCount=spl[1, 'Count']+1)

  # default: install packages with
  # Count>=minCount
  #\dontrun is sometimes run on CRAN. See
  #https://github.com/ThinkR-open/prepare-for-cran
  #accessed 2021-06-14
  if (interactive()) {
    installPackages(spl)
  }

##
## 2. packageSum object
##

splS <- packageSum(spl)
# check the code but do not install anything:
installPackages(splS, splS[1, 'Count']+1)

# install ALL packages
if (interactive()) {
  installPackages(splS, 1)
}
}
```

Description

Obtain a summary by package of a `findFn` object give it class `packageSum`.

This is a simple function, first calling `PackageSum2`, than assigning class `packagesum` to it.

Usage

```
packageSum(x,
  fields=c("Title", "Version", "Author",
    "Maintainer", "Packaged", 'helpPages',
    'vignette', 'URL'),
  lib.loc=NULL, ...)
## S3 method for class 'findFn'
packageSum(x,
  fields=c("Title", "Version", "Author",
    "Maintainer", "Packaged", 'helpPages',
    'vignette', 'URL'),
  lib.loc=NULL, ...)
## S3 method for class 'data.frame'
packageSum(x,
  fields=c("Title", "Version", "Author",
    "Maintainer", "Packaged", 'helpPages',
    'vignette', 'URL'),
  lib.loc=NULL, ...)
## S3 method for class 'list'
packageSum(x,
  fields=c("Title", "Version", "Author",
    "Maintainer", "Packaged", 'helpPages',
    'vignette', 'URL'),
  lib.loc=NULL, ...)
```

Arguments

<code>x</code>	a data.frame with columns <code>Package</code> and <code>Score</code> .
<code>fields</code>	character vector of names of columns to add to <code>x</code>. The function first looks in the components of <code>packageDescription(x\$Package[i])</code>. <code>vignette</code> is obtained via the function of that name. Component <code>Packaged</code> receives special treatment. If present, only the portion preceding <code>;</code> will be retained. This seems to be a time stamp automatically generated by something like R CMD build. It is absent for packages automatically loaded when R is started. In such cases, the third component of <code>strsplit(packageDescription(x\$Package[i])\$Built, ..., ';')</code> will be stored as <code>Packaged</code>. This seems to be a time stamp automatically generated by something like R CMD INSTALL --build.

`lib.loc` an optional `lib.loc` argument passed to `packageDescription`.
`...` additional arguments (currently unused)

Details

With an object of class `findFn`, call [PackageSum2](#), then make it class `packageSum`.

If less than half of the package reference are installed, it prints a note suggesting the user call [installPackages](#), because much of the information is obtained from the packages' DESCRIPTION file.

Value

a `data.frame` of class `c('packageSum', 'data.frame')`.

Author(s)

Spencer Graves

See Also

[findFn](#) [PackageSum2](#) [PackageSummary](#) [installPackages](#)

Examples

```
##
## data.frame method
##
tstdf <- data.frame(Package=c('grid', 'base'),
                    stringsAsFactors=FALSE)
tst2 <- packageSum(tstdf)

##
## list method
##
tstList <- list(PackageSummary=tstdf)

all.equal(tst2, packageSum(tstList))

##
## findFn method
##
tst.findFn <- data.frame(
  Package=c('grid', 'base')[c(1,1,2)],
  Score=2:4, Date=LETTERS[1:3], stringsAsFactors=FALSE)
attr(tst.findFn, 'PackageSummary') <-
  PackageSummary(tst.findFn)
class(tst.findFn) <- c('findFn', 'data.frame')
tst2. <- packageSum(tst.findFn)
```

```

all.equal(tst2, tst2.[names(tst2)])

##
## spline example
##
splineHelp <- findFn("spline", maxPages = 2)
splinePkgs <- packageSum(splineHelp)

```

PackageSum2

Add Info from Installed Packages to PackageSummary

Description

Add information on installed packages to the PackageSummary of a findFn object.

Usage

```

PackageSum2(x,
  fields=c("Title", "Version", "Author", "Maintainer",
    "Packaged", 'helpPages', 'vignette', 'URL'),
  lib.loc=NULL, ...)
## S3 method for class 'findFn'
PackageSum2(x,
  fields=c("Title", "Version", "Author", "Maintainer",
    "Packaged", 'helpPages', 'vignette', 'URL'),
  lib.loc=NULL, ...)
## S3 method for class 'data.frame'
PackageSum2(x,
  fields=c("Title", "Version", "Author", "Maintainer",
    "Packaged", 'helpPages', 'vignette', 'URL'),
  lib.loc=NULL, ...)
## S3 method for class 'list'
PackageSum2(x,
  fields=c("Title", "Version", "Author", "Maintainer",
    "Packaged", 'helpPages', 'vignette', 'URL'),
  lib.loc=NULL, ...)

```

Arguments

<code>x</code>	a data.frame with columns Package and Score.
<code>fields</code>	character vector of names of columns to add to x. The function first looks in the components of packageDescription(x\$Package[i]). 'vignette' is obtained via the function of that name. Component 'Packaged' receives special treatment. If present, only the portion preceding ';' will be retained. This seems to be a time stamp automatically generated by something like R CMD build. It is absent for packages automatically loaded when R is started. In such cases, the third component of

```

        strsplit(packageDescription(x$Package[i])$Built, ..., ';') will be
        stored as 'Packaged'. This seems to be a time stamp automatically generated by
        something like R CMD INSTALL --build.

lib.loc    an optional lib.loc argument passed to packageDescription.
...        additional arguments (currently unused)

```

Details

With an object of class `findFn`, extract the `PackageSummary` attribute and pass it to the `data.frame` method.

With an object of class `list`, extract the `PackageSummary` component and pass it to the `data.frame` method.

For a `data.frame` that is not an `findFn` object, add other columns from attributes of `packageDescription` for installed packages named in the column `Package`. Also, for any packages that are installed, replace the `Date` with the `Packaged` date. The `Date` in Baron's `RSiteSearch` database is the date of acquisition, which will typically be more recent than the `Packaged` date provided the locally installed package has the same version as that in Baron's database. To get the best information from `PackageSum2`, it is wise to first run both [installPackages](#) to ensure that the packages of greatest interest are installed locally and `update.packages()` to make sure you have the latest versions installed locally. Similarly, if `PackageSum2` does not contain complete interest on a package of interest, this can be fixed by installing the package and rerunning `PackageSum2`.

Value

a `data.frame` with additional fields columns appended to a [PackageSummary data.frame](#).

Author(s)

Spencer Graves

See Also

[packageSum](#), which does essentially the same thing but returns an object of class `packageSum`.
[findFn PackageSummary installPackages](#)

Examples

```

##
## data.frame method
##
Tstdf <- data.frame(Package=c('grid', 'base'),
                   stringsAsFactors=FALSE)
Tst2 <- PackageSum2(Tstdf)

##
## list method
##
TstList <- list(PackageSummary=Tstdf)

```



```

all.equal(Tst2, PackageSum2(TstList))

##
## findFn method
##
Tst.findFn <- data.frame(
  Package=c('grid', 'base')[c(1,1,2)],
  Score=2:4, Date=LETTERS[1:3], stringsAsFactors=FALSE)
attr(Tst.findFn, 'PackageSummary') <- PackageSummary(
  Tst.findFn)
class(Tst.findFn) <- c('findFn', 'data.frame')
Tst2. <- PackageSum2(Tst.findFn)

all.equal(Tst2, Tst2.[names(Tst2)])

```

PackageSummary

Summarize findFn Results by Package

Description

Returns a [data.frame](#) with one row for each package and columns Count = number of rows in the search results for that package, maxScore and totalScore = max and total score for help pages found from that package.

Usage

```
PackageSummary(x, sortby=NULL)
```

Arguments

x	a data.frame with columns Package, Score, and Date.
sortby	a character vector specifying how the data.frame returned should be sorted. Default = c('Count', 'MaxScore', 'TotalScore', 'Package') to sort descending on numerics and ascending on alphanumerics. Specifying sortby = c('c', 't', 'm') is equivalent to c('Count', 'TotalScore', 'MaxScore', 'Package'). Components of sortby must match either this list or c('Score', 'Function', 'Date', 'Description', 'Link'). Any on this latter list are ignored without a warning. This allows the same sortby used for findFn to be used here.

Details

1. Convert `x['Package']` to character to automatically drop any unused levels of a factor.
2. Compute `Count`, `TotalScore`, and `MaxScore`.
3. Find the first occurrence of each `Package`, and use that to convert the `Link` to the first help page to `pkgLink` = a link for the package. For example, the `Link` to 'html' for `help('c')` is 'http://finzi.psych.upenn.edu/R/libr' and `pkgLink` to the 'html' overview for 'base' is 'http://finzi.psych.upenn.edu/R/library/base/html/00Index.htm'.
4. Assemble into a `data.frame`, sort and return.

Value

a `data.frame` with one row for each package and columns `Package`, `Count`, `MaxScore`, `TotalScore`, `Date`, and `pkgLink`, sorted as specified by `sortBy`.

Author(s)

Spencer Graves

See Also

`RSiteSearch`, `findFn PackageSum2`, `packageSum`

Examples

```
tstdf <- data.frame(Package=letters[c(1,1,2)], Score=2:4,
                    Date=LETTERS[1:3], stringsAsFactors=FALSE)
tstdf$Link <- paste0('http://finzi.psych.upenn.edu/R/library/',
                    tstdf$Package, '/html/', letters[4:6], '.html')

tstSum <- PackageSummary(tstdf)
# The answer:
tstSm <- data.frame(Package=letters[1:2], Count=c(a=2, b=1),
                    MaxScore=c(3, 4), TotalScore=c(5, 4),
                    Date=LETTERS[c(1, 3)], stringsAsFactors=FALSE)
tstSm$pkgLink <- paste0('http://finzi.psych.upenn.edu/R/library/',
                    tstdf$Package[2:3], '/html/00Index.html')
row.names(tstSm) <- 1:2

all.equal(tstSum, tstSm)
```

`print.findFn`

print a findFn object

Description

Print a `findFn` object to a file and pass it to a web browser

Usage

```
## S3 method for class 'findFn'
print(x, where, title,
      openBrowser = TRUE, template, ...)
```

Arguments

x	An object of class findFn
where	a character vector interpreted as follows: If length(where)==1, it must be either 'HTML' or 'console' or the name of a column of x or the name of a file to hold the file created to be displayed in a web browser. If length(where)>1, it must be the names of columns of x to be displayed on the console. If where is a vector of names of columns of x, those columns will be printed to the console, and there will be no display in a web browser. If where == 'console', the following columns of x are displayed: c('Count', 'Package', 'Function', 'Score', 'Date').
title	An optional title to give the HTML file. Default is to use the original query string.
openBrowser	logical; if TRUE and where is missing or 'HTML', launch default browser after building table
template	Template file used by brew
...	ignored

Value

The full path and name of the file created is returned invisibly.

Author(s)

Sundar Dorai-Raj, Spencer Graves, Romain Francois, Uwe Ligges

See Also

[findFn](#), [RSiteSearch](#), [browseURL](#) [brew](#)

Examples

```
splineSearch <- try(findFn("spline", maxPages = 2))

if(!inherits(splineSearch, 'try-error')){
  if(!CRAN()){
    print(splineSearch, 'console')
    splineSearch # all columns in a browser
  }
webScr <- try(findFn('web scraping'))
if(!inherits(webScr, 'try-error')){
  if(!CRAN()){
```

```

    print(webScr)
  }
}
}

```

print.packageSum	<i>print a packageSum object</i>
------------------	----------------------------------

Description

Print a packageSum object to a file and pass it to a web browser

Usage

```

## S3 method for class 'packageSum'
print(x, where, title,
      openBrowser = TRUE, template, ...)

```

Arguments

x	An object of class packageSum
where	a character vector interpreted as follows: If length(where)==1, it must be either 'HTML' or 'console' or the name of a column of x or the name of a file to hold the file created to be displayed in a web browser. If length(where)>1, it must be the names of columns of x to be displayed on the console. If where is a vector of names of columns of x, those columns will be printed to the console, and there will be no display in a web browser. If where == 'console', the following columns of x are displayed: c('Count', 'maxScore', 'totalScore', 'Package', 'Date').
title	An optional title to give the HTML file. Default is to use the original query string.
openBrowser	logical; if TRUE and where is missing or 'HTML', launch default browser after building table
template	Template file used by brew
...	ignored

Value

The full path and name of the file created is returned invisibly.

Author(s)

Spencer Graves

See Also

[print.findFn](#) [packageSum](#) [findFn](#), [RSiteSearch](#), [browseURL](#) [brew](#)

Examples

```
splineHelp <- try(findFn("spline", maxPages = 2))

if(!inherits(splineHelp, 'try-error')){
  splinePkgs <- packageSum(splineHelp)
  if(!CRAN()){
    print(splinePkgs, 'console')
    splinePkgs # all columns in a browser

    des1 <- try(findFn('differential equations', 1))
    de1 <- try(findFn('differential equation', 1))
    # each retrieves 1 page of 20 hits
    # but not the same 20

    if(!(inherits(des1, 'try-error') ||
      inherits(de1, 'try-error'))){
      de.s <- unionFindFn(des1, de1)
      de.s
    }
  }
  # Other example:
  webScr <- try(findFn('web scraping'))
  if(!inherits(webScr, 'try-error')){
    pS <- packageSum(webScr)
    print(pS)
  }
}
```

sortFindFn

Sort a findFn Object

Description

Sort a [data.frame](#) as a [findFn](#) object.

Usage

```
sortFindFn(x, sortby=NULL)
```

Arguments

x	a data.frame to sort and convert to an object of class findFn (if it does not already have this class).
sortby	sort information as for function findFn .

Details

- 1. pkgSum <- PackageSummary(x, sortby)
- 2. Order x as required for findFn
- 3. class = c("findFn", "data.frame")

Value

An object of class c('findFn', 'data.frame') with a "PackageSummary" attribute.

Author(s)

Spencer Graves

See Also

[findFn sort order](#)

Examples

```
tstdf <- data.frame(Package=letters[c(1,1,2)],
                    Function=c('a1', 'a2', 'b3'), Score=2:4,
                    Date=11:13, Description=c('D1', 'D2', 'D3'),
                    Link=c('L1', 'L2', 'L3'), stringsAsFactors=FALSE)
rss <- sortFindFn(tstdf)
```

summary.findFn	<i>Summary Method for findFn</i>
----------------	----------------------------------

Description

Summary Method for objects of class [findFn](#)

Usage

```
## S3 method for class 'findFn'
summary(object, minPackages = 12,
        minCount = NA, ...)
```

Arguments

object	An object of class findFn
minPackages	the minimum number of packages to include in the summary. Other packages with the same count will also appear in the summary, but packages with a smaller count will not. The number of packages displayed will be less than minPackages only when there are fewer than that number of packages containing the search term in its help pages.

minCount	the minimum count for a package to display. minCount = 1 displays all packages. The default is the minimum of the input minCount and the count for package number minPackages.
...	ignored

Details

Return an object of class `c('summary.findFn', 'list')` with summary information on only packages satisfying the minPackages and minCount criteria. The minPackages and minCount components of the summary output list will be adjusted as necessary to match characteristics of object. The print method for a summary.findFn object will display the minCount, but minPackages will be a component of the returned object without being printed.

Value

An object of class `c('summary.findFn', 'list')` with the following elements:

PackageSummary	a data.frame with one row for each package and columns Package, Count, MaxScore, TotalScore, Date, and pgLink. This summary is sorted per the sortBy argument in the call to findFn.
minPackages, minCount	the minPackages and minCount arguments in this call to summary.findFn.
matches	the total number of matches returned by findFn. This is an attribute of a findFn object; the number of rows of object will either be matches or maxPages*matchesPerPage, whichever is smaller.
nrow	the number of matches in this findFn object
nPackages	the number of packages in this findFn object
call	the matched call to findFn.

Author(s)

Spencer Graves

See Also

[findFn](#), [RSiteSearch](#)

Examples

```
z <- try(findFn("spline", maxPages = 2))
if(!inherits(z, 'try-error'))summary(z, 2)
```

unionFindFn

*Combine findFn Objects***Description**

Combines to [findFn](#) objects into a new findFn object with only one row for any help page duplicated between the two. unionFindFn removes duplicate entries. intersectFindFn keeps only the duplicates.

Usage

```
unionFindFn(e1, e2, sortby=NULL)
intersectFindFn(e1, e2, sortby=NULL)

## S3 method for class 'findFn'
Ops(e1,e2)
# This supports "|" for "unionFindFn"
# and "&" for "intersectFindFn".
```

Arguments

e1, e2	objects of class findFn.
sortby	Optional sortby argument used by sortFindFn and findFn. Default is the sortby argument in attr(e1, 'call').

Details

1. e12 <- rbind(e1, e2)
2. For any (Package, Function) appearing in both e1 and e2, the row with the largest Score is retained and the other is deleted.
3. Apply sortFindFn to the rebuild the summary and sort the result as desired.
4. attr(e12, 'matches') <- c(attr(e1, 'matches'), attr(e2, 'matches'))

Value

an object with class c('findFn', 'data.frame') as returned by sortFindFn and findFn.

Note

Binary operators '&' and '|' are implemented for the S3 class 'findFn'

Author(s)

Spencer Graves and Romain Francois

See Also[findFn](#) [sortFindFn](#)**Examples**

```
des1 <- findFn('differential equations', 1)
de1 <- findFn('differential equation', 1)
# each retrieves 1 page of 20 hits
# but not the same 20

de.s <- unionFindFn(des1, de1)
# combines the two, eliminating duplicates.

# or the sorter version:
de.s. <- des1 | de1

all.equal(de.s, de.s.)

# Keep only the common entries.
de2 <- intersectFindFn(des1, de1)
de2. <- des1 & de1

all.equal(de2, de2.)

# summary and print still work with the combined object.
summary(de.s)
if(!CRAN()){
  de.s
}

summary(de2)
if(!CRAN()){
  de2
}
```

writeFindFn2xls*Write a findFn object to an Excel file*

Description

Write a [findFn](#) object to an Excel file with sheets for [PackageSum2](#), the findFn table, and the call attribute of the findFn object.

Usage

```
writeFindFn2xls(x,
  file.=paste(deparse(substitute(x)), 'xls',
  sep='.'), csv, ...)
findFn2xls(x,
  file.=paste(deparse(substitute(x)), 'xls',
  sep='.'), csv, ...)
```

Arguments

<code>x</code>	An object of class <code>findFn</code>
<code>file.</code>	Name of Excel file to create. If a file of this name already exists, it will be overwritten.
<code>csv</code>	logical: if TRUE, write three *.csv files rather than one *.xls file. Default is FALSE if software is available to write a *.xls file and TRUE otherwise.
<code>...</code>	optional arguments to <code>write.csv</code> used if

Details

`findFn2xls` is an alias for `writeFindFn2xls`; both functions do the same thing.

Value

The name of the file created is returned invisibly.

Author(s)

Spencer Graves with help from Dirk Eddelbuettel, Gabor Grothendiek, and Marc Schwartz.

See Also

[findFn](#), [odbcConnect](#), [sqlSave](#), [odbcClose](#) [WriteXLS](#)

Examples

```
splineSearch <- try(findFn("spline", maxPages = 1))
if(!inherits(splineSearch, 'try-error')){
  writeFindFn2xls(splineSearch)

  findFn2xls(splineSearch, csv=TRUE)
}
```

Index

- * **environment**
 - CRAN, [3](#)
- * **manip**
 - back2ForwardSlash, [2](#)
- * **methods**
 - Extract.findFn, [5](#)
 - print.findFn, [18](#)
 - print.packageSum, [20](#)
 - sortFindFn, [21](#)
 - summary.findFn, [22](#)
- * **misc**
 - findFn, [6](#)
 - grepFn, [9](#)
 - hits, [10](#)
 - PackageSummary, [17](#)
 - unionFindFn, [24](#)
- * **package**
 - installPackages, [11](#)
 - packageSum, [13](#)
 - PackageSum2, [15](#)
- * **print**
 - print.findFn, [18](#)
 - print.packageSum, [20](#)
 - sortFindFn, [21](#)
 - summary.findFn, [22](#)
 - writeFindFn2xls, [25](#)
- ? (findFn), [6](#)
- ??? (findFn), [6](#)
- [.findFn (Extract.findFn), [5](#)
- back2ForwardSlash, [2](#)
- brew, [19](#), [21](#)
- browseURL, [19](#), [21](#)
- CRAN, [3](#)
- data.frame, [5](#), [16–18](#), [21](#)
- download.file, [8](#)
- Extract.findFn, [5](#)
- findFn, [5](#), [6](#), [10–14](#), [16](#), [18](#), [19](#), [21–26](#)
- findFn2xls (writeFindFn2xls), [25](#)
- findFunction, [8](#)
- grep, [10](#)
- grepFn, [9](#)
- gsub, [2](#)
- help.search, [7](#), [8](#)
- hits, [10](#)
- install.packages, [11](#), [12](#)
- installPackages, [11](#), [14](#), [16](#)
- intersectFindFn, [10](#), [11](#)
- intersectFindFn (unionFindFn), [24](#)
- matches (hits), [10](#)
- odbcClose, [26](#)
- odbcConnect, [26](#)
- Ops.findFn (unionFindFn), [24](#)
- order, [22](#)
- packageSum, [12](#), [13](#), [16](#), [18](#), [21](#)
- PackageSum2, [12–14](#), [15](#), [18](#), [25](#)
- PackageSummary, [14](#), [16](#), [17](#)
- print.findFn, [18](#), [21](#)
- print.packageSum, [20](#)
- print.summary.findFn (summary.findFn), [22](#)
- Quotes, [2](#)
- RSiteSearch, [6–8](#), [18](#), [19](#), [21](#), [23](#)
- scan, [2](#)
- skip_on_cran, [4](#)
- sort, [22](#)
- sortFindFn, [21](#), [25](#)
- sqlSave, [26](#)
- summary.findFn, [22](#)

`Sys.getenv`, [4](#)

`unionFindFn`, [10](#), [11](#), [24](#)

`writeFindFn2xls`, [25](#)

`WriteXLS`, [26](#)