Package ‘pkggraph’

November 15, 2018

Type Package

Title A Consistent and Intuitive Platform to Explore the Dependencies of Packages on the Comprehensive R Archive Network Like Repositories

Version 0.2.3

Description

Interactively explore various dependencies of a package(s) (on the Comprehensive R Archive Network Like repositories) and perform analysis using tidy philosophy. Most of the functions return a 'tibble' object (enhancement of 'dataframe') which can be used for further analysis. The package offers functions to produce 'network' and 'igraph' dependency graphs. The 'plot' method produces a static plot based on 'ggnetwork' and 'plot3' function produces an interactive D3 plot based on 'networkD3'.

Imports curl (>= 2.5), dplyr (>= 0.5.0), htmltools (>= 0.3.5), igraph (>= 1.0.1), intergraph (>= 2.0.2), Matrix (>= 1.2.10), networkD3 (>= 0.4), network (>= 1.13.0), RColorBrewer (>= 1.1.2), tibble (>= 1.3.0), tools, utils, plyr (>= 1.8.4)

Depends R (>= 3.5.0), ggnetwork (>= 0.5.1), ggplot2 (>= 2.2.1), data.table (>= 1.10.4)

License GPL-3

Encoding UTF-8

RoxygenNote 6.1.0

Suggests knitr (>= 1.15.1), rmarkdown (>= 1.4), magrittr (>= 1.5), sna (>= 2.4), statnet.common (>= 3.3.0), BiocManager (>= 1.30.4)

VignetteBuilder knitr

URL https://github.com/talegari/pkggraph

BugReports https://github.com/talegari/pkggraph/issues

NeedsCompilation no

Author KS Srikanth [aut, cre], Singh Nikhil [aut]

Maintainer KS Srikanth <sri.teach@gmail.com>

Repository CRAN

Date/Publication 2018-11-15 09:50:03 UTC
Description

Interactively explore various dependencies of a package(s) (on the Comprehensive R Archive Network Like repositories) and perform analysis using tidy philosophy. Most of the functions return a 'tibble' object (enhancement of 'dataframe') which can be used for further analysis. The package offers functions to produce 'network' and 'igraph' dependency graphs. The 'plot' method produces a static plot based on 'ggnetwork' and 'plotd3' function produces an interactive D3 plot based on 'networkD3'.


deptable

Details
See the vignette for further details

Author(s)
Maintainer: KS Srikanth <sri.teach@gmail.com>
Authors:
• Singh Nikhil <nikhilsingh2009@gmail.com>

See Also
Useful links:
• [https://github.com/talegari/pkggraph](https://github.com/talegari/pkggraph)
• Report bugs at [https://github.com/talegari/pkggraph/issues](https://github.com/talegari/pkggraph/issues)

---

deptable
deptable

Description
(tibble) A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'. Every row defines a dependency. This is computed for all packages in 'packmeta'

Usage
deptable

Format
An object of class tbl_df (inherits from tbl, data.frame) with 61154 rows and 3 columns.

---

get_all_dependencies
get_all_dependencies

Description
Get all dependencies

Usage
get_all_dependencies(packages, level = 1L, relation = c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"), strict = FALSE, ignore = c("datasets", "utils", "grDevices", "graphics", "stats", "methods"))
get_all_reverse_dependencies

Arguments

packages (non-empty character vector) Package names
level (positive integer, Default = 1L) Depth of recursive dependency
relation (character vector) Types of relations. Must be a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")
strict (logical, Default: TRUE) Whether to consider all packages (alternately only 'relation' specific packages) when computing dependencies for the next level
ignore package names to ignore

Value

A tibble with three columns: ‘pkg_1’, ‘relation’ and ‘pkg_2’

Author(s)

Srikanth KS

See Also

get_all_reverse_dependencies

Examples

pkggraph::init(local = TRUE)
# general use
pkggraph::get_all_dependencies("mlr")
# specify two levels
pkggraph::get_all_dependencies("mlr", level = 2)
# specify relation(s)
pkggraph::get_all_dependencies("mlr", level = 2, relation = "Imports")
# setting strict to TRUE to only consider 'Imports' of the previous level
pkggraph::get_all_dependencies("mlr",
  , level = 2
  , relation = "Imports"
  , strict = TRUE)

get_all_reverse_dependencies

Description

Get all reverse dependencies
Usage

get_all_reverse_dependencies(packages, level = 1L,
  relation = c("Depends", "Imports", "LinkingTo", "Suggests",
  "Enhances"), strict = FALSE, ignore = c("datasets", "utils",
  "grDevices", "graphics", "stats", "methods"))

Arguments

- packages (non-empty character vector) Package names
- level (positive integer, Default = 1L) Depth of recursive dependency
- relation (character vector) Types of relations. Must be a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")
- strict (logical, Default: TRUE) Whether to consider all packages (alternately only 'relation' specific packages) when computing dependencies for the next level
- ignore package names to ignore

Value

A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'

Author(s)

Srikanth KS

See Also

get_all_dependencies

Examples

pkggraph::init(local = TRUE)
# general use
pkggraph::get_all_reverse_dependencies("mlr")
# specify two levels
pkggraph::get_all_reverse_dependencies("mlr", level = 2)
# specify relation(s)
pkggraph::get_all_reverse_dependencies("mlr", level = 2, relation = "Imports")
# setting strict to TRUE to only consider 'Imports' of the previous level
pkggraph::get_all_reverse_dependencies("mlr",
  level = 2,
  relation = "Imports",
  strict = TRUE)
Description
Get dependencies

Usage
get_depends(packages, level = 1L)

Arguments
packages (non-empty character vector) Package names
level (positive integer) Depth of recursive dependency

Value
A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'

Author(s)
Srikanth KS

See Also
get_depends, get_imports, get_linkingto, get_suggests, get_enhances, get_all_dependencies, get_reverse_depends

Examples
pkggraph::init(local = TRUE)
pkggraph::get_depends("glmnet")

Description
Get dependencies

Usage
get_enhances(packages, level = 1L)
### get_imports

**Arguments**
- **packages** *(non-empty character vector)* Package names
- **level** *(positive integer)* Depth of recursive dependency

**Value**
A tibble with three columns: `pkg_1`, `relation` and `pkg_2`

**Author(s)**
Srikanth KS

**See Also**
get_deps, get_imports, get_linkingto, get_suggests, get_enhances, get_all_dependencies, get_reverse_enhances

**Examples**
```r
cgg::init(local = TRUE)
cgg::get_enhances("bigmemory")
```

---

### Description
Get dependencies

### Usage
```r
get_imports(packages, level = 1L)
```

**Arguments**
- **packages** *(non-empty character vector)* Package names
- **level** *(positive integer)* Depth of recursive dependency

**Value**
A tibble with three columns: `pkg_1`, `relation` and `pkg_2`

**Author(s)**
Srikanth KS
get_linkingto

See Also

get_depends, get_imports, get_linkingto, get_suggests, get_enhances, get_all_dependencies, get_reverse_imports

Examples

pkggraph::init(local = TRUE)
pkggraph::get_imports("dplyr")

given_dependencies <- get_linkingto(packages, level = 1L)

Arguments

packages (non-empty character vector) Package names
level (positive integer) Depth of recursive dependency

Value

A tibble with three columns: ‘pkg_1’, ‘relation’ and ‘pkg_2’

Author(s)

Srikanth KS

See Also

get_depends, get_imports, get_linkingto, get_suggests, get_enhances, get_all_dependencies, get_reverse_linkingto

Examples

pkggraph::init(local = TRUE)
pkggraph::get_linkingto("tibble")
get_neighborhood

Description

Obtain dependencies and reverse dependencies of packages at a given depth of recursion

Usage

get_neighborhood(packages, level = 1L, relation = c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances"), strict = FALSE, interconnect = TRUE, ignore = c("datasets", "utils", "grDevices", "graphics", "stats", "methods"))

Arguments

packages (non-empty character vector) Package names
level (positive integer, Default: 1L) Depth of recursive dependency
relation (character vector) Types of relations. Must be a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")
strict (logical, Default: TRUE) Whether to consider all packages (alternately only 'relation' specific packages) when computing dependencies for the next level
interconnect (flag, Default: TRUE) Whether to capture dependency among packages (of a given level) of the next level (See examples)
ignore package names to ignore

Value

A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'

Author(s)

Srikanth KS

See Also

neighborhood_graph, make_neighborhood_graph

Examples

# explore first level dependencies
pkggraph::init(local = TRUE)
pkggraph::get_neighborhood("caret")

# explore second level dependencies
pkggraph::get_neighborhood("caret", level = 2)
get_reverse_depends

Description
Get reverse dependencies

Usage
get_reverse_depends(packages, level = 1L)

Arguments
packages (non-empty character vector) Package names
level (positive integer) Depth of recursive dependency

Value
A tibble with three columns: ‘pkg_1’, ‘relation’ and ‘pkg_2’

Author(s)
Srikanth KS

See Also
get_reverse_depends, get_reverse_imports, get_reverse_linkingto, get_reverse_suggests, get_reverse_enhances, get_all_reverse_dependencies, get_depends
**get_reverse_enhances**

**Examples**

```r
pkggraph::init(local = TRUE)
pkggraph::get_reverse_depends("utils")
```

---

**Description**

Get reverse dependencies

**Usage**

```r
get_reverse_enhances(packages, level = 1L)
```

**Arguments**

- `packages` (non-empty character vector): Package names
- `level` (positive integer): Depth of recursive dependency

**Value**

A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'

**Author(s)**

Srikanth KS

**See Also**

- `get_reverse_depends`, `get_reverse_imports`, `get_reverse_linkingto`, `get_reverse_suggests`, `get_reverse_enhances`, `get_all_reverse_dependencies`, `get_enhances`

**Examples**

```r
pkggraph::init(local = TRUE)
pkggraph::get_reverse_enhances("synchronicity")
```
get_reverse_imports

Description
Get reverse dependencies

Usage
get_reverse_imports(packages, level = 1L)

Arguments
packages (non-empty character vector) Package names
level (positive integer) Depth of recursive dependency

Value
A tibble with three columns: `pkg_1`, `relation` and `pkg_2`

Author(s)
Srikanth KS

See Also
get_reverse_departs, get_reverse_imports, get_reverse_linkingto, get_reverse_suggests,
get_reverse_enhances, get_all_reverse_dependencies, get_imports

Examples
pkggraph::init(local = TRUE)
pkggraph::get_reverse_imports("Rcpp")

get_reverse_linkingto

Description
Get reverse dependencies

Usage
get_reverse_linkingto(packages, level = 1L)
get_reverse_suggests

Arguments

- packages: (non-empty character vector) Package names
- level: (positive integer) Depth of recursive dependency

Value

A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'

Author(s)

Srikanth KS

See Also

get_reverse_depends, get_reverse_imports, get_reverse_linkingto, get_reverse_suggests, get_reverse_enhances, get_all_reverse_dependencies, get_linkingto

Examples

pkggraph::init(local = TRUE)
pkggraph::get_reverse_linkingto("BH")

Description

Get reverse dependencies

Usage

get_reverse_suggests(packages, level = 1L)

Arguments

- packages: (non-empty character vector) Package names
- level: (positive integer) Depth of recursive dependency

Value

A tibble with three columns: 'pkg_1', 'relation' and 'pkg_2'

Author(s)

Srikanth KS
### Description

Get dependencies

### Usage

```r
get_suggests(packages, level = 1L)
```

### Arguments

- **packages** (non-empty character vector) Package names
- **level** (positive integer) Depth of recursive dependency

### Value

A tibble with three columns: ‘pkg_1’, ‘relation’ and ‘pkg_2’

### Author(s)

Srikanth KS

### See Also

- `get_depends`
- `get_imports`
- `get_linkingto`
- `get_suggests`
- `get_enhances`
- `get_all_dependencies`
- `get_reverse_suggests`

### Examples

```r
pkggraph::init(local = TRUE)
pkggraph::get_reverse_suggests("purrr")
```
**init**

**Description**

Initiate the package by loading the data into parent frame. This should be done as soon as the package is loaded or attached. This creates (rewrites) new variables 'deptable' and 'packmeta' to the environment where it is run from.

**Usage**

```r
init(local = FALSE, repository = "CRAN", ...)
```

**Arguments**

- `local` (flag, default: FALSE) If
  - FALSE: Tries to download package data from CRAN over internet and compute dependencies
  - TRUE: Loads data that comes with the package corresponding to 2nd September 2017 02:04 IST

- `repository` (character vector, Default: "CRAN") One among c("CRAN", "BioCsoft", "BioCann", "BioCexp", "BioCextra", "omegahat"). To use a repository not in this list, set 'repository' to NULL and pass named argument called 'repos' with a valid repository address. This will be passed as is to `utils::available.packages()`.

... Additional parameters to be passed to 'available.packages()

**Value**

An invisible TRUE

**Author(s)**

Srikanth KS

---

**make_neighborhood_graph**

**Description**

Make a network or igraph graph object of dependencies and reverse dependencies from tibble output by functions like 'get_neighborhood', 'get_all_dependents' etc
Usage

make_neighborhood_graph(ndf, type = "igraph")

Arguments

ndf (tibble) Output by functions like 'get_neighborhood', 'get_all_dependents' etc
type (string, Default: "igraph") Graph object type. Either "network" or "igraph"

Value

A network or igraph graph object

Author(s)

Srikanth KS

See Also

neighborhood_graph, get_neighborhood

Examples

pkggraph::init(local = TRUE)
graph_object <- pkggraph::get_neighborhood("caret")
pkggraph::make_neighborhood_graph(graph_object)

Description

Obtain a network or igraph graph object of dependencies and reverse dependencies of packages at
a given depth of recursion

Usage

neighborhood_graph(packages, level = 1L, type = "igraph",
relation = c("Depends", "Imports", "LinkingTo", "Suggests",
"Enhances"), strict = FALSE, interconnect = TRUE,
ignore = c("datasets", "utils", "grDevices", "graphics", "stats",
"methods"))
neighborhood_graph

Arguments

packages (non-empty character vector) Package names
level (positive integer, Default: 1L) Depth of recursive dependency
type (string, Default: "igraph") Graph object type. Either "network" or "igraph"
relation (character vector) Types of graph edges. Must be a subset of c("Depends", "Imports", "LinkingTo", "Suggests", "Enhances")
strict (logical, Default: TRUE) Whether to consider all packages (alternately only 'relation' specific packages) when computing dependencies for the next level
interconnect (flag, Default: TRUE) Whether to capture dependency among packages (of a given level) of the next level (See examples)
ignore package names to ignore

Value

A network or igraph graph object

Author(s)

Srikanth KS

See Also

get_neighborhood, make_neighborhood_graph

Examples

# explore first level dependencies
pkggraph::init(local = TRUE)
pkggraph::neighborhood_graph("caret")

# explore second level dependencies of class network
pkggraph::neighborhood_graph("caret", level = 2, type = "network")

# get 'imports' specific neighborhood of 'mlr' package with strict = TRUE
neighborhood_graph("mlr"
  , level = 2
  , strict = TRUE
  , interconnect = FALSE
  , relation = "Imports")

# get 'imports' specific neighborhood of 'mlr' package with strict = FALSE
neighborhood_graph("mlr"
  , level = 2
  , strict = FALSE
  , interconnect = FALSE
  , relation = "Imports")
### packmeta

**Description**

(A character matrix) Output of `utils::available.packages`

**Usage**

`packmeta`

**Format**

An object of class `matrix` with 11328 rows and 17 columns.

### plot.pkggraph

**Description**

plot a `pkggraph` object

**Usage**

```r
## S3 method for class 'pkggraph'
plot(x, ...)```

**Arguments**

- **x**: plot object generated by `neighborhood_graph` or `make_neighborhood_graph`
- **...**: additional arguments (See details)

**Details**

- `background`: "black" or "white". Default is ‘black’
- `nodeImportance`: "in", "out" or "both", in - Node will be considered important(and increased size) if more incoming. out - Node will be considered important if more outgoing. both - Node importance will be calculated on both incoming and outgoing. True for all the nodes. Default is ‘both’
- `edgeLabel`: logical. TRUE if edge label has to be shown. Default is FALSE

**Author(s)**

Nikhil Singh
See Also

neighborhood_graph, make_neighborhood_graph, get_neighborhood

Examples

```r
## Not run:
pkggraph::init(local = TRUE)
plot_obj <- pkggraph::neighborhood_graph("hash")
plot(plot_obj)

plot_obj <- pkggraph::neighborhood_graph("tidytext")
plot(plot_obj,
     background = "white"
     nodeImportance = "out")
plot_obj <- pkggraph::neighborhood_graph(c("hash" ,"tokenizers")
     , interconnect = FALSE
     )

plot(plot_obj, background = "white")

## End(Not run)
```

Description

D3 network of a pkggraph object

Usage

```r
plotd3(x, height = 500, width = 1000)
```

Arguments

- `x`: plot object generated by `neighborhood_graph` or `make_neighborhood_graph` of type igraph
- `height`: parameter to change the height of the d3 plot. Default is 500
- `width`: parameter to change the width of the d3 plot. Default is 1000

Author(s)

Nikhil Singh
Examples

```r
## Not run:
pkggraph::init(local = TRUE)
plot_obj <- pkggraph::neighborhood_graph("hash")
 pkggraph::plotd3(plot_obj)

plot_obj <- pkggraph::neighborhood_graph(c("hash", "tidytext"))
 pkggraph::plotd3(plot_obj, height = 750, width = 1200)

plot_obj <- pkggraph::neighborhood_graph(c("hash", "Matrix"))
 pkggraph::plotd3(plot_obj)

## End(Not run)
```

Description

Captures recursive dependencies of these types: "Depends", "Imports", "LinkingTo"

Usage

`relies(packages)`

Arguments

- `packages` (non-empty character vector) Package names

Value

(Named list) A name is the package name from 'packages'. A Value is a character vector of all packages which the package 'relies' (Captures recursive dependencies of these types: "Depends", "Imports", "LinkingTo")

Author(s)

Srikanth KS

See Also

`reverse_relies`

Examples

```r
pkggraph::init(local = TRUE)
pkggraph::relies("tidytext")
```
reverse_relies

Description
Captures reverse recursive dependencies of these types: "Depends", "Imports", "LinkingTo"

Usage
reverse_relies(packages)

Arguments
packages (non-empty character vector) Package names

Value
(Named list) A name is the package name from 'packages'. A Value is a character vector of all packages which the package 'relies' (Captures reverse recursive dependencies of these types: "Depends", "Imports", "LinkingTo")

Author(s)
Srikanth KS

See Also
reliess

Examples
pkggraph::init(local = TRUE)
pkggraph::reverse_relies("data.table")

Check depends

Description
Check whether pkg_1 has a dependency on pkg_2

Usage
pkg_1 %depends% pkg_2
Arguments

pkg_1 a package name
pkg_2 a package name

Value

TRUE or FALSE

Author(s)

Srikanth KS

Examples

pkggraph::init(local = TRUE)
"dplyr" %depends% "tibble"

Description

Check whether pkg_1 has a dependency on pkg_2

Usage

pkg_1 %enhances% pkg_2

Arguments

pkg_1 a package name
pkg_2 a package name

Value

TRUE or FALSE

Author(s)

Srikanth KS

Examples

pkggraph::init(local = TRUE)
"dplyr" %enhances% "tibble"
## %imports%

*Check imports*

**Description**

Check whether pkg_1 has a dependency on pkg_2

**Usage**

```r
pkg_1 %imports% pkg_2
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pkg_1</td>
<td>a package name</td>
</tr>
<tr>
<td>pkg_2</td>
<td>a package name</td>
</tr>
</tbody>
</table>

**Value**

TRUE or FALSE

**Author(s)**

Srikanth KS

**Examples**

```r
pkggraph::init(local = TRUE)
"dplyr" %imports% "tibble"
```

## %linkingto%

*Check linkingto*

**Description**

Check whether pkg_1 has a dependency on pkg_2

**Usage**

```r
pkg_1 %linkingto% pkg_2
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pkg_1</td>
<td>a package name</td>
</tr>
<tr>
<td>pkg_2</td>
<td>a package name</td>
</tr>
</tbody>
</table>
Value
TRUE or FALSE

Author(s)
Srikanth KS

Examples
pkggraph::init(local = TRUE)
"dplyr" %linkingto% "tibble"

Description
Check whether a package has a recursive dependency on the other

Usage
pkg_1 %relies% pkg_2

Arguments
pkg_1 (string) A package name
pkg_2 (string) A package name

Value
(flag) TRUE, if 'pkg_1' 'relies' on 'pkg_2'

Author(s)
Srikanth KS

See Also
relies, reverse_relies

Examples
pkggraph::init(local = TRUE)
"dplyr" %relies% "tibble"
\textbf{%suggests%} \hspace{1cm} \textit{Check suggests}

\section*{Description}
Check whether \texttt{pkg\_1} has a dependency on \texttt{pkg\_2}

\section*{Usage}
\texttt{pkg\_1 \%suggests\% pkg\_2}

\section*{Arguments}
\begin{itemize}
  \item \texttt{pkg\_1} \hspace{.5cm} a package name
  \item \texttt{pkg\_2} \hspace{.5cm} a package name
\end{itemize}

\section*{Value}
TRUE or FALSE

\section*{Author(s)}
Srikanth KS

\section*{Examples}
\begin{verbatim}
pkggraph::init(local = TRUE)
"dplyr" \%suggests\% "tibble"
\end{verbatim}
Index

*Topic datasets
  deptable, 3
  packmeta, 18
%depends%, 21
%enhances%, 22
%imports%, 23
%linkingto%, 23
%relies%, 24
%suggests%, 25
deptable, 3
get_all_dependencies, 3, 5–8, 14
get_all_reverse_dependencies, 4, 4, 10–14
get_depends, 6, 6, 7, 8, 10, 14
get_enhances, 6, 6, 7, 8, 11, 14
get_imports, 6, 7, 7, 8, 12, 14
get_linkingto, 6–8, 8, 13, 14
get_neighborhood, 9, 16, 17, 19
get_reverse_depends, 6, 10, 10, 11–14
get_reverse_enhances, 7, 10, 11, 11, 12–14
get_reverse_imports, 8, 10–12, 12, 13, 14
get_reverse_linkingto, 8, 10–12, 12, 13, 14
get_reverse_suggests, 10–13, 13, 14
get_suggests, 6–8, 14, 14
init, 15
make_neighborhood_graph, 9, 15, 17–19
neighborhood_graph, 9, 16, 16, 18, 19
packmeta, 18
pkggraph (pkggraph-package), 2
pkggraph-package, 2
plot.pkggraph, 18
plotd3, 19
relies, 20, 21, 24
reverse_relies, 20, 21, 24