

# Package ‘pkggraph’

June 2, 2017

**Type** Package

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

**Version** 0.2.0

## 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'.

**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 (>= 3.4.0), utils (>= 3.4.0)

**Depends** R (>= 3.4.0), ggnetwork (>= 0.5.1), ggplot2 (>= 2.2.1)

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 6.0.1

**Suggests** knitr (>= 1.15.1), rmarkdown (>= 1.4), magrittr (>= 1.5), sna (>= 2.4), statnet.common (>= 3.3.0), tidyverse (>= 1.1.1), BiocInstaller (>= 1.24.0)

**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** 2017-06-02 06:28:25 UTC

**R topics documented:**

|  |           |
|--|-----------|
| pkggraph-package . . . . .             | 2         |
| deptable . . . . .                     | 3         |
| get_all_dependencies . . . . .         | 3         |
| get_all_reverse_dependencies . . . . . | 4         |
| get_depends . . . . .                  | 6         |
| get_enhances . . . . .                 | 6         |
| get_imports . . . . .                  | 7         |
| get_linkingto . . . . .                | 8         |
| get_neighborhood . . . . .             | 9         |
| get_reverse_depends . . . . .          | 10        |
| get_reverse_enhances . . . . .         | 11        |
| get_reverse_imports . . . . .          | 12        |
| get_reverse_linkingto . . . . .        | 12        |
| get_reverse_suggests . . . . .         | 13        |
| get_suggests . . . . .                 | 14        |
| init . . . . .                         | 15        |
| make_neighborhood_graph . . . . .      | 15        |
| neighborhood_graph . . . . .           | 16        |
| packmeta . . . . .                     | 18        |
| plot . . . . .                         | 18        |
| plotd3 . . . . .                       | 19        |
| relies . . . . .                       | 20        |
| reverse_relies . . . . .               | 21        |
| %depends% . . . . .                    | 21        |
| %enhances% . . . . .                   | 22        |
| %imports% . . . . .                    | 23        |
| %linkingto% . . . . .                  | 23        |
| %relies% . . . . .                     | 24        |
| %suggests% . . . . .                   | 25        |
| <b>Index</b>                           | <b>26</b> |

---

|                  |                 |
|------------------|-----------------|
| pkggraph-package | <i>pkggraph</i> |
|------------------|-----------------|

---

**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'.

**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>
- Report bugs at <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 55699 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"))
```

**Arguments**

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

---

|             |                    |
|-------------|--------------------|
| get_depends | <i>get_depends</i> |
|-------------|--------------------|

---

**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()  
pkggraph::get_depends("glmnet")
```

---

|              |                     |
|--------------|---------------------|
| get_enhances | <i>get_enhances</i> |
|--------------|---------------------|

---

**Description**

Get dependencies

**Usage**

```
get_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\\_depends](#), [get\\_imports](#), [get\\_linkingto](#), [get\\_suggests](#), [get\\_enhances](#), [get\\_all\\_dependencies](#), [get\\_reverse\\_enhances](#)

**Examples**

```
pkggraph::init()  
pkggraph::get_enhances("bigmemory")
```

---

|                          |                    |
|--------------------------|--------------------|
| <code>get_imports</code> | <i>get_imports</i> |
|--------------------------|--------------------|

---

**Description**

Get dependencies

**Usage**

```
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

**See Also**

[get\\_depends](#), [get\\_imports](#), [get\\_linkingto](#), [get\\_suggests](#), [get\\_enhances](#), [get\\_all\\_dependencies](#), [get\\_reverse\\_imports](#)

**Examples**

```
pkggraph::init()
pkggraph::get_imports("dplyr")
```

---

|                            |                      |
|----------------------------|----------------------|
| <code>get_linkingto</code> | <i>get_linkingto</i> |
|----------------------------|----------------------|

---

**Description**

Get dependencies

**Usage**

```
get_linkingto(packages, level = 1L)
```

**Arguments**

|                       |  |
|-----------------------|--|
| <code>packages</code> | (non-empty character vector) Package names       |
| <code>level</code>    | (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()
pkggraph::get_linkingto("tibble")
```



---

get\_neighborhood      *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()
pkggraph::get_neighborhood("caret")

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

```
# explore second level dependencies without
# considering dependencies from third level
pkggraph::get_neighborhood("caret", level = 2, interconnect = FALSE)

# explore first level dependencies of multiple packages
# and consider second level dependencies
get_neighborhood(c("caret", "mlr"))

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

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

---

`get_reverse_depends`    *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](#)

### Examples

```
pkggraph::init()
pkggraph::get_reverse_depends("utils")
```

---

`get_reverse_enhances` *get\_reverse\_enhances*

---

### Description

Get reverse dependencies

### Usage

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

### Arguments

|                       |  |
|-----------------------|--|
| <code>packages</code> | (non-empty character vector) Package names       |
| <code>level</code>    | (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

```
pkggraph::init()
pkggraph::get_reverse_enhances("synchronicity")
```

---

`get_reverse_imports`    *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\\_depends](#), [get\\_reverse\\_imports](#), [get\\_reverse\\_linkingto](#), [get\\_reverse\\_suggests](#),  
[get\\_reverse\\_enhances](#), [get\\_all\\_reverse\\_dependencies](#), [get\\_imports](#)

**Examples**

```
pkggraph::init()  
pkggraph::get_reverse_imports("Rcpp")
```

---

`get_reverse_linkingto`    *get\_reverse\_linkingto*

---

**Description**

Get reverse dependencies

**Usage**

```
get_reverse_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\\_reverse\\_depends](#), [get\\_reverse\\_imports](#), [get\\_reverse\\_linkingto](#), [get\\_reverse\\_suggests](#),  
[get\\_reverse\\_enhances](#), [get\\_all\\_reverse\\_dependencies](#), [get\\_linkingto](#)

**Examples**

```
pkggraph::init()  
pkggraph::get_reverse_linkingto("BH")
```

---

`get_reverse_suggests` *get\_reverse\_suggests*

---

**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

**See Also**

[get\\_reverse\\_depends](#), [get\\_reverse\\_imports](#), [get\\_reverse\\_linkingto](#), [get\\_reverse\\_suggests](#), [get\\_reverse\\_enhances](#), [get\\_all\\_reverse\\_dependencies](#), [get\\_suggests](#)

**Examples**

```
pkggraph::init()
pkggraph::get_reverse_suggests("purrr")
```

---

|              |                     |
|--------------|---------------------|
| get_suggests | <i>get_suggests</i> |
|--------------|---------------------|

---

**Description**

Get dependencies

**Usage**

```
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**

```
pkggraph::init()
pkggraph::get_suggests("knitr")
```

---

 init

*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

```
init(local = TRUE, repository = "CRAN", ...)
```

### Arguments

|            |   |
|------------|---|
| local      | (flag, default: TRUE) If <ul style="list-style-type: none"> <li>• FALSE: Tries to to download package data from CRAN over internet and compute dependencies</li> <li>• TRUE: Loads data that comes with the package corresponding to 1st June 2017 17:14 IST</li> </ul>   |
| 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

*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()  
graph_object <- pkggraph::get_neighborhood("caret")  
pkggraph::make_neighborhood_graph(graph_object)
```

---

neighborhood\_graph     *neighborhood\_graph*

---

**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"))
```



**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()
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 | <i>packmeta</i> |
|----------|-----------------|

---

**Description**

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

**Usage**

packmeta

**Format**

An object of class `matrix` with 10718 rows and 17 columns.

---

|      |                               |
|------|-------------------------------|
| plot | <i>plot a pkggraph object</i> |
|------|-------------------------------|

---

**Description**

plot a `pkggraph` object

**Usage**

plot(x, ...)

**Arguments**

|     |  |
|-----|--|
| x   | plot object generated by <a href="#">neighborhood_graph</a> or <a href="#">make_neighborhood_graph</a> |
| ... | 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**

```
## Not run:
pkggraph::init()
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)
```

---

plotd3

*plotd3*

---

**Description**

D3 network of a pkggraph object

**Usage**

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

**Arguments**

|        |   |
|--------|---|
| x      | plot object generated by <a href="#">neighborhood_graph</a> or <a href="#">make_neighborhood_graph</a> 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**

```
## Not run:
pkggraph::init()
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)
```

---

relies

*relies*

---

**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**

```
pkggraph::init()
pkggraph::relies("tidytext")
```

---

|                |                       |
|----------------|-----------------------|
| reverse_relies | <i>reverse_relies</i> |
|----------------|-----------------------|

---

**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**

[relies](#)

**Examples**

```
pkggraph::init()
pkggraph::reverse_relies("data.table")
```

---

|           |                      |
|-----------|----------------------|
| %depends% | <i>Check depends</i> |
|-----------|----------------------|

---

**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()  
"dplyr" %depends% "tibble"
```

---

%enhances%

*Check enhances*

---

**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()  
"dplyr" %enhances% "tibble"
```

---

%imports%                      *Check imports*

---

**Description**

Check whether pkg\_1 has a dependency on pkg\_2

**Usage**

pkg\_1 %imports% pkg\_2

**Arguments**

pkg\_1                      a package name  
pkg\_2                      a package name

**Value**

TRUE or FALSE

**Author(s)**

Srikanth KS

**Examples**

```
pkggraph::init()  
"dplyr" %imports% "tibble"
```

---

%linkingto%                      *Check linkingto*

---

**Description**

Check whether pkg\_1 has a dependency on pkg\_2

**Usage**

pkg\_1 %linkingto% pkg\_2

**Arguments**

pkg\_1                      a package name  
pkg\_2                      a package name

**Value**

TRUE or FALSE

**Author(s)**

Srikanth KS

**Examples**

```
pkggraph::init()
"dplyr" %linkingto% "tibble"
```

---

%relies%

*Check relies*

---

**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()
"dplyr" %relies% "tibble"
```



---

%suggests%                      *Check suggests*

---

### **Description**

Check whether pkg\_1 has a dependency on pkg\_2

### **Usage**

```
pkg_1 %suggests% pkg_2
```

### **Arguments**

pkg\_1                      a package name  
pkg\_2                      a package name

### **Value**

TRUE or FALSE

### **Author(s)**

Srikanth KS

### **Examples**

```
pkggraph::init()  
"dplyr" %suggests% "tibble"
```

# 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, [18](#)

plotd3, [19](#)

relies, [20](#), [21](#), [24](#)

reverse\_relies, [20](#), [21](#), [24](#)