

Package ‘deps’

July 22, 2025

Type Package

Title Dependency Management with 'roxygen'-Style Comments

Version 0.4.0

Description Manage your source code dependencies
by decorating your existing R code with special,
'roxygen'-style comments.

License MIT + file LICENSE

LazyLoad yes

Imports renv, jsonlite, remotes

RoxygenNote 7.3.1

Encoding UTF-8

BugReports <https://github.com/analythium/deps/issues>

URL <https://hub.analythium.io/deps/>,
<https://github.com/analythium/deps>

Language en-US

NeedsCompilation no

Author Peter Solymos [aut, cre] (ORCID:
<<https://orcid.org/0000-0001-7337-1740>>),
Analythium Solutions Inc. [cph, fnd]

Maintainer Peter Solymos <peter@analythium.io>

Repository CRAN

Date/Publication 2024-07-07 15:00:02 UTC

Contents

create	2
install	3
Index	4

`create`*Create a Dependencies JSON File*

Description

Discover dependencies and write a `dependencies.json` file.

Usage

```
create(  
  dir = getwd(),  
  file = "dependencies.json",  
  output = dir,  
  installed = c("base", "recommended"),  
  overwrite = TRUE,  
  ask = TRUE  
)
```

Arguments

<code>dir</code>	Path to the directory where the files to be scanned are located.
<code>file</code>	The name of the file to be save, default is <code>"dependencies.json"</code> .
<code>output</code>	Path to the directory where JSON file should be written to.
<code>installed</code>	The priority argument for <code>installed.packages()</code> for packages to be excluded.
<code>overwrite</code>	Logical, should the file in the output directory be overwritten if exists?
<code>ask</code>	Logical, asking confirmation before writing the <code>dependencies.json</code> file.

Value

Invisibly returns the list of file names that were created. The side effect is a JSON (and possibly a text for system requirements) file written to the hard drive. The function fails when there are no R related files in `dir`.

Examples

```
dir <- system.file("examples/01-basic", package = "deps")  
out <- tempdir()  
create(dir, output = out, ask = interactive())  
cat(readLines(file.path(out, "dependencies.json")), sep = "\n")  
unlink(file.path(out, "dependencies.json"))
```

`install`*Install Dependencies*

Description

Install dependencies from an existing `dependencies.json` file or after discovering the dependencies.

Usage

```
install(  
  dir = getwd(),  
  file = "dependencies.json",  
  upgrade = "never",  
  cleanup = TRUE,  
  timeout = 300L,  
  ask = TRUE,  
  ...  
)
```

Arguments

<code>dir</code>	Path to the directory where the JSON file should be written to.
<code>file</code>	The name of the file to be save, default is <code>"dependencies.json"</code> . If the file is not found in <code>dir</code> , <code>create()</code> is called.
<code>upgrade</code>	Should package dependencies be upgraded? Argument passed to <code>remotes</code> functions.
<code>cleanup</code>	Logical, clean up files created by <code>create()</code> when file does not exist.
<code>timeout</code>	Integer, timeout for file downloads (default 60 seconds can be short).
<code>ask</code>	Logical, asking confirmation before writing the <code>dependencies.json</code> file.
<code>...</code>	Other argument passed to <code>remotes</code> functions.

Value

Returns `NULL` invisibly. The side effect is the dependencies installed.

Examples

```
dir <- system.file("examples/01-basic", package = "deps")  
out <- tempdir()  
create(dir, output = out, ask = interactive())  
cat(readLines(file.path(out, "dependencies.json")), sep = "\n")  
## Not run:  
install(out)  
  
## End(Not run)  
unlink(file.path(out, "dependencies.json"))
```

Index

create, [2](#)

install, [3](#)