

# Package ‘maketools’

March 28, 2023

**Type** Package

**Title** Exploring and Testing the Toolchain and System Libraries

**Version** 1.2.4

**Description** A collection of helper functions that interface with the appropriate system utilities to learn about the build environment. Lets you explore 'make' rules to test the local configuration, or query 'pkg-config' to find compiler flags and libs needed for building packages with external dependencies. Also contains tools to analyze which libraries that a installed R package linked to by inspecting output from 'ldd' in combination with information from your distribution package manager, e.g. 'rpm' or 'dpkg'. Finally the package provides Windows-specific utilities to automatically find or install the suitable version of the 'Rtools' build environment, and diagnose some common problems.

**License** MIT + file LICENSE

**URL** <https://github.com/jeroen/maketools>

**Encoding** UTF-8

**Imports** sys (>= 3.1)

**RoxygenNote** 7.2.1

**VignetteBuilder** knitr

**Suggests** curl, knitr, rmarkdown, testthat

**Language** en-US

**NeedsCompilation** no

**Author** Jeroen Ooms [aut, cre, cph] (<<https://orcid.org/0000-0002-4035-0289>>)

**Maintainer** Jeroen Ooms <jeroen@berkeley.edu>

**Repository** CRAN

**Date/Publication** 2023-03-28 21:20:02 UTC

## R topics documented:

|             |   |
|-------------|---|
| diagnostics | 2 |
| find_logo   | 2 |

|                     |          |
|---------------------|----------|
| make . . . . .      | 3        |
| pkgconfig . . . . . | 4        |
| rtools . . . . .    | 5        |
| r_config . . . . .  | 6        |
| sysdeps . . . . .   | 7        |
| <b>Index</b>        | <b>8</b> |

---

|             |                           |
|-------------|---------------------------|
| diagnostics | <i>Diagnostics Report</i> |
|-------------|---------------------------|

---

### Description

Print some diagnostics about your compiler environment. These are also shown when the `make tools` package is attached.

### Usage

```
maketools_diagnostics()
```

### See Also

Other maketools: [make\(\)](#), [pkgconfig](#), [r\\_config](#), [rtools](#), [sysdeps](#)

---

|           |                      |
|-----------|----------------------|
| find_logo | <i>Package tools</i> |
|-----------|----------------------|

---

### Description

Get some extra info about packages.

### Usage

```
find_logo(path = ".")
```

### Arguments

|      |                           |
|------|---------------------------|
| path | root directory of package |
|------|---------------------------|

---

make

*Make*

---

## Description

Compile C / C++ / Fortran source files using the compiler configured by your R Makeconf file.

## Usage

```
make(target = "all", makefile = r_makeconf_path())
```

```
make_call(cmd = "$(CC)", args = "--version")
```

```
make_echo(cmd = "$(CC)")
```

```
make_info()
```

## Arguments

|          |   |
|----------|---|
| target   | name of output file that you want to make   |
| makefile | path to the Makefile. Defaults to the Makeconf which R uses when building R packages. |
| cmd      | command to invoke (may be a variable)   |
| args     | additional arguments for cmd  |

## Details

The make function literally calls `make yourfile.o -f /path/to/R/Makeconf`. This is exactly what R does when building packages and hence the best way to test if the compiler is working.

## See Also

Other maketools: [diagnostics](#), [pkgconfig](#), [r\\_config](#), [rtools](#), [sysdeps](#)

## Examples

```
# Test the CXX compiler
if(cxx_info()$available){
  testprog <- '#include <iostream>\nint main() {std::cout << "Hello World!";}
writeLines(testprog, con = 'testprog.cc')
make('testprog')

# Test and cleanup
system('./testprog')
unlink('testprog*', recursive = TRUE)
}

# Run a program from a make variable
```

```
make_call('${CXX}', '--version')

# Where your makeconf is stored:
make_info()
```

---

pkgconfig

*Query pkg-config*

---

## Description

Wrappers for the pkg-config utility to query information on C/C++ libraries that are available on your system.

## Usage

```
pc_info()

pc_pkg_list()

pc_pkg_exists(pkg = "libcurl")

pc_pkg_version(pkg = "libcurl")

pc_pkg_cflags(pkg = "libcurl")

pc_pkg_libs(pkg = "libcurl", static = FALSE)

pc_pkg_info(pkg = "libcurl")
```

## Arguments

|        |  |
|--------|--|
| pkg    | names of the pkg-config libraries to query             |
| static | get libs for static linking, i.e. include dependencies |

## See Also

Other maketools: [diagnostics](#), [make\(\)](#), [r\\_config](#), [rtools](#), [sysdeps](#)

## Examples

```
# Check if pkg-config is available
(info <- pc_info())
if(info$available)
  pc_pkg_list()
```

## Description

Tools to test if a suitable version of Rtools is available, or help the user to set this up.

## Usage

```
rtools_find()
```

```
rtools_registry_info()
```

```
rtools_install(silent = TRUE)
```

## Arguments

`silent` performs automatic unattended installation with all the default options. If set to `FALSE` the user has to click through the usual installation wizard.

## Details

Unlike most operating systems, Windows does not include a native compiler. Hence in order to build R packages with compiled C/C++/Fortran code on Windows, you need to install our custom toolchain bundle called Rtools.

There are currently 2 versions of Rtools available:

- [rtools40](#): required for compiling packages on R-4.0 and newer
- [rtools35](#): required for compiling packages on R-3.6 and older

The function [rtools\\_find](#) shows information about a suitable version of Rtools installed on your system. If needed, it automatically adds `make` to the `PATH` of the current session. If `rtools_find()` returns a list (containing toolchain information), this means everything is ready to install packages from source using [install.packages](#) and others. It returns `NULL` if no suitable version is found on the system.

The [rtools\\_install](#) function automatically downloads and installs the appropriate version of Rtools for your current version of R.

## See Also

Other maketools: [diagnostics](#), [make\(\)](#), [pkgconfig](#), [r\\_config](#), [sysdeps](#)

---

r\_config

*R CMD Config*

---

## Description

Cross-platform wrappers for R CMD config to lookup the availability of the compiler.

## Usage

cc\_info()

cxx\_info()

cxx11\_info()

cxx14\_info()

cxx17\_info()

fc\_info()

r\_cmd\_config(VAR = "--all")

## Arguments

VAR           value passed to R CMD config such as CXX or FC

## See Also

Other maketools: [diagnostics](#), [make\(\)](#), [pkgconfig](#), [rtools](#), [sysdeps](#)

## Examples

```
# This runs 'R CMD CONFIG CXX'
r_cmd_config("CXX")

# Show C++ config:
cxx_info()
```

**Description**

Shows the external shared libraries that an installed R package is linked to by running `ldd` on the package `so` file. Then uses system package manager (e.g. `dpkg` or `rpm` or `brew`) to locate which system package that contains the binaries, headers, and (if available) sources for this library.

**Usage**

```
package_sysdeps(pkg, lib.loc = NULL)
```

```
package_sysdeps_string(pkg, lib.loc = NULL)
```

```
package_links_to(pkg, lib.loc = NULL)
```

**Arguments**

|                      |  |
|----------------------|--|
| <code>pkg</code>     | name of an installed R package                   |
| <code>lib.loc</code> | path to the R package directory for this package |

**Details**

For common distributions, the output also includes a URL to the distro-homepage of the system package. Here we can typically find more information about the package, such as configuration options, dependencies, and custom patches applied by your distribution.

Because we use `ldd`, this only shows run-time dependencies of an installed R package. This is especially relevant if you distribute the compiled R package in binary form, because the same external libraries need to be available on the user/deployment machine. This tool does not show dependencies that are only needed at build-time, such as static or header-only libraries, and other utilities required to build the package.

**See Also**

Other maketools: [diagnostics](#), [make\(\)](#), [pkgconfig](#), [r\\_config](#), [rtools](#)

# Index

## \* maketools

- diagnostics, 2
  - make, 3
  - pkgconfig, 4
  - r\_config, 6
  - rtools, 5
  - sysdeps, 7
- cc\_info (r\_config), 6
- cxx11\_info (r\_config), 6
- cxx14\_info (r\_config), 6
- cxx17\_info (r\_config), 6
- cxx\_info (r\_config), 6
- diagnostics, 2, 3–7
- fc\_info (r\_config), 6
- find\_logo, 2
- install.packages, 5
- make, 2, 3, 4–7
- make\_call (make), 3
- make\_echo (make), 3
- make\_info (make), 3
- maketools\_diagnostics (diagnostics), 2
- package\_links\_to (sysdeps), 7
- package\_sysdeps (sysdeps), 7
- package\_sysdeps\_string (sysdeps), 7
- pc\_info (pkgconfig), 4
- pc\_pkg\_cflags (pkgconfig), 4
- pc\_pkg\_exists (pkgconfig), 4
- pc\_pkg\_info (pkgconfig), 4
- pc\_pkg\_libs (pkgconfig), 4
- pc\_pkg\_list (pkgconfig), 4
- pc\_pkg\_version (pkgconfig), 4
- pkgconfig, 2, 3, 4, 5–7
- r\_cmd\_config (r\_config), 6
- r\_config, 2–5, 6, 7
- rtools, 2–4, 5, 6, 7
- rtools\_find, 5
- rtools\_find (rtools), 5
- rtools\_install, 5
- rtools\_install (rtools), 5
- rtools\_registry\_info (rtools), 5
- sysdeps, 2–6, 7