Package ‘MakefileR’
August 29, 2016

Encoding UTF-8
Title Create ‘Makefiles’ Using R
Description A user-friendly interface for the construction of ‘Makefiles’.
Version 1.0
Date 2016-01-08
Imports magrittr
Suggests testthat, knitr, rmarkdown
License GPL-3
LazyData true
URLNote https://github.com/krlmlr/MakefileR
URL http://krlmlr.github.io/MakefileR
BugReports https://github.com/krlmlr/MakefileR/issues
VignetteBuilder knitr
RoxygenNote 5.0.1.9000
NeedsCompilation no
Author Kirill Müller [aut, cre]
Maintainer Kirill Müller <krlmr+r@mailbox.org>
Repository CRAN
Date/Publication 2016-01-08 15:55:12

R topics documented:
c.MakefileR_group .......................................................... 2
makefile ................................................................. 2
make_comment ........................................................... 3
make_def ................................................................. 4
make_group ............................................................. 5
make_rule ............................................................... 6
make_text ............................................................... 7
write_makefile ......................................................... 8
### Description

Rules can be appended to groups and Makefiles using the `c` function or the `+` operator.

### Usage

```c
## S3 method for class 'MakefileR_group'
c(..., recursive = FALSE)

## S3 method for class 'MakefileR_group'
x + y
```

### Arguments

- `x`, `y`: Rules, the first (or the first element of ...) must be of class `MakefileR_group` (created by `make_group` or `makefile`)
- `recursive`: [any]
- `null`: Unused

### Examples

```c
makefile()
makefile() + (make_group() + make_comment("Definitions") + make_def("A", "a"))
```

### Description

A Makefile consists of a list of rules, definition, comments and other items.

### Usage

```c
makefile(..., .dots = NULL)
```
make_comment

Arguments

... [MakefileR]
   Items created by make_rule or other make_functions
dots [list]
   Further rules in addition to ...

Details

Use the c function or the + operator to append rules, definitions, comments, plain text, and groups.

Value

An object of class MakefileR_file

References

https://www.gnu.org/software/make/manual/

See Also

make_rule, make_def, make_comment, make_text, make_group, c.MakefileR_group

Examples

makefile(make_rule("all", c("first_target", "second_target")))

---------

make_comment  Creates a Makefile comment

---------

Description

For helping the reader understand what’s happening

Usage

make_comment(...)
make_def

Value
An object of class MakefileR_comment

References
https://www.gnu.org/software/make/manual/

See Also
Other items: make_def, make_group, make_rule, make_text

Examples
make_comment("This is a comment")

---

make_def Creates a variable definition in a Makefile

Description
A variable definition in a Makefile consists of a variable name and its definition. Both are separated by the equality sign =.

Usage
make_def(variable, definition)

Arguments
variable [character(1)]
Variable name
definition [character(1)]
Definition for this variable

Details
No quoting is applied to the definition by this function. Currently, both variable and definition are required to be character values of length one.

Use the c function or the + operator to append definitions to groups and Makefiles.

Value
An object of class MakefileR_def

References
https://www.gnu.org/software/make/manual/
See Also

`makefile`, `make_group`

Other items: `make_comment`, `make_group`, `make_rule`, `make_text`

Examples

```r
make_def("R_USER_LIBRARY", .libPaths()[[1L]])
makefile() +
  make_def("R_USER_LIBRARY", .libPaths()[[1L]])
```

---

### Description

Helps separating similar rules.

### Usage

```r
make_group(..., .dots = NULL, sep = NULL)
```

### Arguments

- `...` [MakefileR]
  - Items created by `make_rule` or other `make_` functions
- `.dots` [list]
  - Further rules in addition to `...`
- `sep` [character(1)]
  - Separator between group items, `NULL` (the default) means no separator.

### Details

Use the `c` function or the `+` operator to append groups to other groups and Makefiles (thus creating nested groups).

### Value

An object of class `MakefileR_group`

### References


### See Also

`c.MakefileR_group`

Other items: `make_comment`, `make_def`, `make_rule`, `make_text`
make_rule

Creates a Makefile rule

Description

A rule in a Makefile consists of a (list of) targets which may depend on one or more dependencies each. Optionally, a script is executed to create the target. Generally, multiple targets mean that the rule is identical for each of the individual targets, and multiple dependencies mean that all of them are required to build each of the targets. In the script, the target can be referenced by $@, and the first dependency can be referenced by $<. Note that the dollar sign has a special meaning in a Makefile, use $ in scripts that need to use the dollar sign themselves.

Usage

make_rule(targets, deps = NULL, script = NULL)

Arguments

targets [character]
Target names

deps [character]
Dependency names

script [character]
A script to execute to build the targets.

Details

Use the c function or the + operator to append rules to groups and Makefiles.

Value

An object of class MakefileR_rule

References

https://www.gnu.org/software/make/manual/

See Also

makefile

Other items: make_comment, make_def, make_group, make_text
**make_text**  

Creates a custom Makefile entry

---

**Examples**

```make
make_rule("all", c("first_target", "second_target"))
make_rule(".FORCE")
make_rule("first_target", ".FORCE", "echo 'Building first target'")
make_rule("second_target", "first_target",
c("echo 'Building second target''', "echo 'Done'"))
```

```make
makefile() +
make_rule("all", c("first_target", "second_target")) +
make_rule(".FORCE") +
make_rule("first_target", ".FORCE", "echo 'Building first target'") +
make_rule("second_target", "first_target",
c("echo 'Building second target''', "echo 'Done'"))
```

---

**Description**

For anything else not covered by the other make_ functions, such as the export statement for exporting Makefile variables.

**Usage**

```make
make_text(...)  
```

**Arguments**

```make
...  [character]
Custom text
```

**Details**

Use the c function or the + operator to append comments to groups and Makefiles.

**Value**

An object of class MakefileR_text

**References**

https://www.gnu.org/software/make/manual/

**See Also**

Other items: make_comment, make_def, make_group, make_rule
write_makefile

Examples

make_text("export SOME_VARIABLE")

Description

Makefiles, as created by `makefile`, only exist in memory until they are written to a file by this function.

Usage

`write_makefile(makefile, file_name)`

Arguments

- **makefile** [MakefileR]
  A Makefile, created by `makefile`
- **file_name** [character(1)]
  Target file name

Value

The value returned by `writeLines`
Index

+. 2–7
+. MakefileR_group (c.MakefileR_group), 2

c. 2–7
c.MakefileR_group, 2, 3, 5

make_comment, 3, 3, 5–7
make_def, 3, 4, 4, 5–7
make_group, 2–5, 5, 6, 7
make_rule, 3–5, 6, 7
make_text, 3–6, 7
makefile, 2, 2, 5, 6, 8

write_makefile, 8
writeLines, 8