

Package ‘rolog’

January 12, 2023

Type Package

Title Query 'SWI'-'Prolog' from R

Version 0.9.10

Date 2023-01-10

Maintainer Matthias Gondan <Matthias.Gondan-Rochon@uibk.ac.at>

Description This R package connects to 'SWI'-
'Prolog', <<https://www.swi-prolog.org/>>, so that R can send deterministic and non-
deterministic queries to 'prolog' ('consult', 'query'/'submit', 'once', 'findall').

License FreeBSD

Imports Rcpp (>= 1.0.7), methods, utils

Depends R (>= 4.2)

URL <https://github.com/mgondan/rolog>

BugReports <https://github.com/mgondan/rolog/issues>

LinkingTo Rcpp, rswipl

RoxygenNote 7.2.3

Encoding UTF-8

SystemRequirements GNU Make, swi-prolog

Suggests rmarkdown, knitr, rswipl, DiagrammeR, DiagrammeRsvg, rsvg,
htmltools, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr, rmarkdown

NeedsCompilation yes

Author Matthias Gondan [aut, com, cre] (Universität Innsbruck)

Repository CRAN

Date/Publication 2023-01-12 12:00:02 UTC

R topics documented:

as.rolog	2
clear	3
consult	3
findall	4
once	5
portray	7
postproc	8
preproc	8
query	9
rolog_done	10
rolog_init	11
rolog_options	11
submit	12
Index	13

as.rolog	<i>Translate simplified to canonical representation</i>
----------	---------------------------------------------------------

Description

Translate simplified to canonical representation

Usage

```
as.rolog(query = quote(member(.X, "[a, "b", 3L, 4, (pi), TRUE, .Y])))
```

Arguments

query	an R call representing a Prolog query with prolog-like syntax, e.g., ‘member(.X, "[a, b, .Y]’ for use in [query()], [once()], and [findall()]. The argument is translated to Rolog’s representation, with R calls corresponding to Prolog terms and R expressions corresponding to Prolog variables. Variables and expressions in parentheses are evaluated.
-------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

See Also

[query()], [once()], [findall()]

Examples

```
q <- quote(member(.X, "[a, "b", 3L, 4, pi, (pi), TRUE, .Y]))
as.rolog(q)
```

```
q <- quote(member(.X, "[a, "b", 3L, 4, pi, (pi), TRUE, .Y]))
findall(as.rolog(q))
```

clear	<i>Clear current query</i>
-------	----------------------------

Description

Clear current query

Usage

```
clear()
```

Value

TRUE (invisible)

See Also

[query\(\)](#) for a opening a query.

[submit\(\)](#) for a submitting a query.

[once\(\)](#) for a opening a query, submitting it, and clearing it again.

[findall\(\)](#) for a opening a query, collecting all solutions, and clearing it again.

Examples

```
query(call("member", expression(X), list(quote(a), "b", 3L, 4)))
submit() # X = a
submit() # X = "b"
clear()
```

consult	<i>Consult a prolog database</i>
---------	----------------------------------

Description

Consult a prolog database

Usage

```
consult(fname = system.file(file.path("pl", "family.pl"), package = "rolog"))
```

Arguments

fname file name of database

Value

TRUE on success

See Also

[once\(\)](#), [findall\(\)](#), and [query\(\)/submit\(\)/clear\(\)](#) for executing queries

Examples

```
consult(fname=system.file(file.path("pl", "family.pl"), package="rolog"))
findall(call("ancestor", quote(pam), expression(X)))
```

findall	<i>Invoke a query several times</i>
---------	-------------------------------------

Description

Invoke a query several times

Usage

```
findall(
  query = call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, expression(Y))),
  options = list(portray = FALSE),
  env = globalenv()
)
```

Arguments

query	an R call. The R call consists of symbols, integers and real numbers, character strings, boolean values, expressions, lists, and other calls. Vectors of booleans, integers, floating point numbers, and strings with length $N > 1$ are translated to prolog compounds $!/N$, $%/N$, $#/N$ and $$$/N$, respectively. The names can be modified with the options below.
options	This is a list of options controlling translation from and to prolog. <ul style="list-style-type: none"> • <i>boolvec</i> (see option <code>rolog.boolvec</code>, default is <code>!</code>) is the name of the prolog compound for vectors of booleans. • <i>intvec</i>, <i>realvec</i>, <i>charvec</i> define the compound names for vectors of integers, doubles and strings, respectively (defaults are <code>%</code>, <code>#</code> and <code>\$\$</code>). • If <i>scalar</i> is TRUE (default), vectors of length 1 are translated to scalar prolog elements. If <i>scalar</i> is FALSE, vectors of length 1 are also translated to compounds.
env	The R environment in which the query is run (default: <code>globalenv()</code>). This is mostly relevant for <code>r_eval/2</code> .

Value

If the query fails, an empty list is returned. If the query succeeds $N \geq 1$ times, a list of length N is returned, each element being a list of conditions for each solution, see [once\(\)](#).

See Also

[once\(\)](#) for a single query

[query\(\)](#), [submit\(\)](#), and [clear\(\)](#) for fine-grained control over non-deterministic queries

[rolog_options\(\)](#)

Examples

```
# This query returns a list stating that it works if X = a, "b", ...
findall(call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, NULL, NA)))

# Continued
findall(call("member", expression(X), list(call("sin", call("/", quote(pi), 2)), expression(Y))))

# The same using simplified syntax
q <- quote(member(.X, "[a, "b", 3L, 4, TRUE, NULL, NA, sin(pi/2), .Y])
findall(as.rolog(q))
```

once

Invoke a query once

Description

Invoke a query once

Usage

```
once(
  query = call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, expression(Y))),
  options = list(portray = FALSE),
  env = globalenv()
)
```

Arguments

query	an R call. The R call consists of symbols, integers and real numbers, character strings, boolean values, expressions, lists, and other calls. Vectors of booleans, integers, floating point numbers, and strings with length $N > 1$ are translated to prolog compounds $!N$, $\%N$, $\#N$ and $\$N$, respectively. The names can be modified with the options below.
options	This is a list of options controlling translation from and to prolog.

- *boolvec* (see option `rolog.boolvec`, default is `!`) is the name of the prolog compound for vectors of booleans.
- *intvec*, *realvec*, *charvec* define the compound names for vectors of integers, doubles and strings, respectively (defaults are `%`, `#` and `$$`).
- If *scalar* is `TRUE` (default), vectors of length 1 are translated to scalar prolog elements. If *scalar* is `FALSE`, vectors of length 1 are also translated to compounds.

`env` The R environment in which the query is run (default: `globalenv()`). This is mostly relevant for `r_eval/2`.

Value

If the query fails, `FALSE` is returned. If the query succeeds, a (possibly empty) list is returned that includes the bindings required to satisfy the query.

See Also

[findall\(\)](#) for querying all solutions

[query\(\)](#), [submit\(\)](#), and [clear\(\)](#) for fine-grained control over non-deterministic queries

[rolog_options\(\)](#) for options controlling R to prolog translation

Examples

```
# This query returns FALSE
once(call("member", 1, list(quote(a), quote(b), quote(c))))

# This query returns an empty list meaning yes, it works
once(call("member", 3, list(1, 2, 3)))

# This query returns a list stating that it works if X = 1
once(call("member", 1, list(quote(a), expression(X))))

# The same query using simplified syntax
q = quote(member(1, "[a, .X]"))
once(as.rolog(q))

# This query returns a list stating that X = 1 and Z = expression(Y)
once(call("=", list(expression(X), expression(Y)), list(1, expression(Z))))

# This works for X = [1 | _]; i.e. something like [1](1, expression(_6330))
once(call("member", 1, expression(X)))

# This returns S = '1.0' (scalar)
once(call("format", call("string", expression(S)), "~w", list(1)), options=list(scalar=TRUE))

# This returns S = '#(1.0)' (vector), because the 1 is translated to #(1.0).
# To prevent "~w" from being translated to $$("~w"), it is given as an atom.
once(call("format", call("string", expression(S)), as.symbol("~w"), list(1)),
      options=list(scalar=FALSE))
```

 portray

Translate an R call to a prolog compound and pretty print it

Description

Translate an R call to a prolog compound and pretty print it

Usage

```
portray(
  query = call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, expression(Y))),
  options = NULL
)
```

Arguments

query	an R call. The R call consists of symbols, integers and real numbers, character strings, boolean values, expressions and lists, and other calls. Vectors of booleans, integers, floating point numbers, and strings with length $N > 1$ are translated to prolog compounds $!/N$, $%/N$, $#/N$ and $$$/N$, respectively. The names can be modified with the options below.
options	This is a list of options controlling translation from and to prolog. <ul style="list-style-type: none"> • <i>boolvec</i> (see option <code>rolog.boolvec</code>, default is <code>!</code>) is the name of the prolog compound for vectors of booleans. • <i>intvec</i>, <i>realvec</i>, <i>charvec</i> define the compound names for vectors of integers, doubles and strings, respectively (defaults are <code>%</code>, <code>#</code> and <code>\$\$</code>). • If <i>scalar</i> is <code>TRUE</code> (default), vectors of length 1 are translated to scalar prolog elements. If <i>scalar</i> is <code>FALSE</code>, vectors of length 1 are also translated to compounds.

Details

The R elements are translated to the following prolog citizens:

- numeric -> real (vectors of size N -> $#/N$)
- integer -> integer (vectors -> $%/N$)
- character -> string (vectors -> $$$/N$)
- symbol/name -> atom
- expression -> variable
- call/language -> compound
- boolean -> true, false (atoms)
- list -> list

Value

character string with the prolog syntax of the call

See Also

[rolog_options\(\)](#) for fine-grained control over the translation

postproc	<i>Default hook for postprocessing</i>
----------	----------------------------------------

Description

Default hook for postprocessing

Usage

```
postproc(constraint = call("=<", 1, 2))
```

Arguments

`constraint` the R call representing constraints of the Prolog query.

Value

The default hook translates the inequality and smaller-than-or-equal-to back from Prolog (`\=`, `=<`) to R (`!=`, `<=`).

See Also

[\[rolog_options\(\)\]](#) for fine-grained control over the translation

preproc	<i>Default hook for preprocessing</i>
---------	---------------------------------------

Description

Default hook for preprocessing

Usage

```
preproc(query = quote(1 <= sin))
```

Arguments

`query` the R call representing the Prolog query.

Value

The default hook translates the inequality and smaller-than-or-equal-to from R (\neq , \leq) to Prolog (\neq , \leq). Moreover, primitive functions are converted to regular functions.

See Also

[`rolog_options()`] for fine-grained control over the translation

query	<i>Create a query</i>
-------	-----------------------

Description

Create a query

Usage

```
query(
  query = call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, expression(Y))),
  options = NULL,
  env = globalenv()
)
```

Arguments

query	an R call. The R call consists of symbols, integers and real numbers, character strings, boolean values, expressions, lists, and other calls. Vectors of booleans, integers, floating point numbers, and strings with length $N > 1$ are translated to prolog compounds $!N$, $\%N$, $\#N$ and $\$N$, respectively. The names can be modified with the options below.
options	This is a list of options controlling translation from and to prolog. <ul style="list-style-type: none"> • <i>boolvec</i> (see option <code>rolog.boolvec</code>, default is <code>!</code>) is the name of the prolog compound for vectors of booleans. • <i>intvec</i>, <i>realvec</i>, <i>charvec</i> define the compound names for vectors of integers, doubles and strings, respectively (defaults are <code>%</code>, <code>#</code> and <code>\$\$</code>). • If <i>scalar</i> is <code>TRUE</code> (default), vectors of length 1 are translated to scalar prolog elements. If <i>scalar</i> is <code>FALSE</code>, vectors of length 1 are also translated to compounds.
env	The R environment in which the query is run (default: <code>globalenv()</code>). This is mostly relevant for <code>r_eval/2</code> .

Details

SWI-Prolog does not allow multiple open queries. If another query is open, it is closed and a warning is shown.

Value

If the creation of the query succeeds, TRUE.

See Also

[once\(\)](#) for a query that is submitted only a single time.

[findall\(\)](#) for a query that is submitted until it fails.

Examples

```
query(call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, expression(Y))))
submit() # X = a
submit() # X = "b"
clear()
```

```
query(call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, expression(Y),
  NA, NaN, Inf, NULL, function(x) {y <- sin(x); y^2})))
submit() # X = a
submit() # X = "b"
submit() # X = 3L
submit() # X = 4.0
submit() # X = TRUE
submit() # X = expression(Y) or Y = expression(X)
submit() # X = NA
submit() # X = NaN
submit() # X = Inf
submit() # X = NULL
submit() # X = function(x) {y <- sin(x); y^2})
submit() # FALSE (no more results)
submit() # warning that no query is open
```

```
query(call("member", expression(X), list(quote(a), "b", 3L, 4)))
query(call("member", expression(X), list(TRUE, expression(Y)))) # warning that another query is open
clear()
```

 rolog_done

Clean up when detaching the library

Description

Clean up when detaching the library

Usage

```
rolog_done()
```

Value

‘TRUE’ on success

rolog_init *Start prolog*

Description

Start prolog

Usage

`rolog_init(argv1 = commandArgs()[1])`

Arguments

`argv1` file name of the R executable

Details

SWI-prolog is automatically initialized when the rolog library is loaded, so this function is normally not directly invoked.

Value

'TRUE' on success

rolog_options *Quick access the package options*

Description

Quick access the package options

Usage

`rolog_options()`

Details

Translation from R to Prolog

- numeric vector of size N -> *realvec/N* (default is #)
- integer vector of size N -> *intvec/N* (default is %)
- boolean vector of size N -> *boolvec/N* (default is !)
- character vector of size N -> *charvec/N* (default is \$\$)
- *scalar*: if TRUE (default), translate R vectors of length 1 to scalars
- *portray*: if TRUE (default) whether to return the prolog translation as an attribute to the return value of `once()`, `query()` and `findall()`

Value

list with some options for translating R expressions to prolog

submit

Submit a query that has been opened with `query()` before.

Description

Submit a query that has been opened with `query()` before.

Usage

```
submit(options = NULL)
```

Arguments

`options` This is a list of options controlling translation from and to Prolog. Here, only *postproc* is relevant.

Value

If the query fails, FALSE is returned. If the query succeeds, a (possibly empty) list is returned that includes the bindings required to satisfy the query.

See Also

`query()` for a opening a query.

`rolog_options()` for fine-grained control on the translation from R to Prolog and back.

`clear()` for a clearing a query.

`once()` for a opening a query, submitting it, and clearing it again.

`findall()` for a opening a query, collecting all solutions, and clearing it again.

Examples

```
query(call("member", expression(X), list(quote(a), "b", 3L, 4, expression(Y))))
submit() # X = 3L
submit() # X = 4.0
submit() # X = TRUE
submit() # X = expression(Y) or Y = expression(X)
submit() # FALSE
submit() # warning that no query is open

query(call("member", expression(X), list(quote(a), "b", 3L, 4)))
submit() # X = a
submit() # X = "b"
clear()
```

Index

as.rolog, 2

clear, 3

clear(), 4–6, 12

consult, 3

findall, 4

findall(), 3, 4, 6, 10–12

once, 5

once(), 3–5, 10–12

portray, 7

postproc, 8

preproc, 8

query, 9

query(), 3–6, 11, 12

rolog_done, 10

rolog_init, 11

rolog_options, 11

rolog_options(), 5, 6, 8, 12

submit, 12

submit(), 3–6