Package 'rolog'

November 12, 2022

Type Package

Title Query 'SWI'-'Prolog' from R

Version 0.9.6

Date 2022-10-21

Author Matthias Gondan

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)

Depends R (>= 4.2), utils

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

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

LinkingTo Rcpp, rswipl

RoxygenNote 7.2.0

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

Repository CRAN

Date/Publication 2022-11-12 20:50:02 UTC

R topics documented:

`as.rolog` *Translate simplified to canonical representation*

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 *Clear current query*

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 *Consult a prolog database*

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` *Invoke a query several times*

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)
)
```

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.

Value

If the query fails, an empty list is returned. If the query succeeds $N >= 1$ 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)
)
```

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.

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, 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 $$/N, respectively. The names can be modified with the options below.
`options`	This is a list of options controlling translation from and to prolog. • *boolvec* (see option `rolog.boolvec`, default is !) is the name of the prolog compound for vectors of booleans. • *intvec*, *realvec*, *charvec* define the compound names for vectors of integers, doubles and strings, respectively (defaults are %, # and $$). • 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.

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

query *Create a query*

Description

Create a query

Usage

```
query(
  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, 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.

Details

SWI-Prolog does not allow multiple open queries. If another query is open, it 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()

# The same in simplified syntax
q <- quote(member(.X, ""[a, "b", 3L, 4, TRUE, .Y]))
query(as.rolog(q))
submit() # X = a
submit() # X = "b"
clear()

query(call("member", expression(X), list(quote(a), "b", 3L, 4, TRUE, 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)))
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 of 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()
```

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.

`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