

Package ‘rLingo’

March 31, 2018

Type Package

Title R Interface to LINGO API

Version 18.0.1

Date 2018-03-02

Author Zhe Liu, Kevin Cunningham

Maintainer Zhe Liu<zliu@lindo.com>

Depends R (>= 2.14.1)

Description An R interface to LINGO API functions, which gives you the ability to access all the major features of LINGO. To download the trial version LINGO, please visit www.lindo.com/rlingo.

SystemRequirements LINGO 18.0

URL www.lindo.com/rlingo

License LGPL (>= 2.1)

Collate rLingo.R zzz.R rLingoParam.R

Repository CRAN

NeedsCompilation yes

R topics documented:

rLingo	2
rLSclearPointersLng	3
rLScloseLogFileLng	4
rLScreateEnvLicenseLng	4
rLScreateEnvLng	5
rLSdeleteEnvLng	5
rLSexecuteScriptLng	6
rLSopenLogFileLng	6
rLSparam	7
rLSsetCharPointerLng	7
rLSsetDouPointerLng	8
rLSsetIntPointerLng	9

Index	10
--------------	-----------

Description

R interface to LINGO API functions. For more information, please refer to LINGO User Manual.

Details

In R interface all function names use the convention of 'r' + LINGO API function name. E.g, function rLScreateEnvLng in R corresponds to LScreateEnvLng in LINGO API.

References

LINDO SYSTEMS home page at www.lindo.com

Examples

```
#The "Simple" example, which can be found in folder "Programming Samples" of the
#installation of Lingo 15.0.
```

```
#load the package
library(rLingo)
```

```
#create Lingo environment object
rEnv = rLScreateEnvLng()
```

```
#open LINGO's log file
rLSopenLogFileLng(rEnv, "Simple.log")
```

```
#pass memory transfer pointers to LINGO
#define pnPointersNow
pnPointersNow = integer(1)
```

```
#@POINTER(1)
dProfit = c(100., 150.)
rLSsetDouPointerLng(rEnv, dProfit, pnPointersNow)
pnPointersNow
```

```
#@POINTER(2)
dLimit = c(100., 120.)
rLSsetDouPointerLng(rEnv, dLimit, pnPointersNow)
pnPointersNow
```

```
#@POINTER(3)
dLabor = c(1., 2.)
rLSsetDouPointerLng(rEnv, dLabor, pnPointersNow)
pnPointersNow
```

```
#@POINTER(4)
dObjective = numeric(1)
rLSsetDouPointerLng(rEnv, dObjective, pnPointersNow)
pnPointersNow
```

```

#@POINTER(5)
dStatus = numeric(1)
rLSsetDouPointerLng(rEnv, dStatus, pnPointersNow)
pnPointersNow

#@POINTER(6)
dProduce = numeric(2)
rLSsetDouPointerLng(rEnv, dProduce, pnPointersNow)
pnPointersNow

#@POINTER(7)
cComputers = "STANDARD\nTURBO\n"
rLSsetCharPointerLng(rEnv, cComputers, pnPointersNow)
pnPointersNow

#Here is the script we want LINGO to run
cScript = "SET ECHOIN 1 \n TAKE SIMPLE.LNG \n GO \n QUIT \n"

#Run the script
rLSexecuteScriptLng(rEnv, cScript)

#Close the log file
rLScloseLogFileLng(rEnv)

#check solution
dStatus
dObjective
dProduce

#delete Lingo enviroment object
rLSdeleteEnvLng(rEnv)

```

rLSclearPointersLng	<i>Clears out the list of @POINTER() pointers to user memory transfer areas established through calls to rLSsetDouPointerLng, rLSsetIntPointerLng, or rLSsetCharPointerLng.</i>
---------------------	---

Description

R interface function for LINGO API function LSclearPointersLng. For more information, please refer to LINGO User Manual.

Usage

```
rLSclearPointersLng(env)
```

Arguments

env	A LINGO environment object.
-----	-----------------------------

Value

An R list object with components:

ErrorCode	Zero if successful, nonzero otherwise.
-----------	--

References

LINDO SYSTEMS home page at www.lindo.com

rLScloseLogFileLng	<i>Closes LINGO's log file that was opened previously by a call to rLSopenLogFileLng.</i>
--------------------	---

Description

R interface function for LINGO API function LScloseLogFileLng. For more information, please refer to LINGO User Manual.

Usage

```
rLScloseLogFileLng(env)
```

Arguments

env	A LINGO environment object.
-----	-----------------------------

Value

An R list object with components:

ErrorCode	Zero if successful, nonzero otherwise.
-----------	--

References

LINDO SYSTEMS home page at www.lindo.com

rLScreeateEnvLicenseLng	<i>Create a LINGO environment object by specifying a license key.</i>
-------------------------	---

Description

R interface function for LINGO API function LScreeateEnvLicenseLng. For more information, please refer to LINGO User Manual.

Usage

```
rLScreeateEnvLicenseLng(pcLicenseKey)
```

Arguments

pcLicenseKey	A text string containing a LINGO license key.
--------------	---

Value

If successful, rLScreeateEnvLicenseLng returns a LINGO environment object; Otherwise, it returns NULL.

References

LINDO SYSTEMS home page at www.lindo.com

See Also

[rLScrtateEnvLng](#) [rLScrtateEnvLng](#)

rLScrtateEnvLng	<i>Create a LINGO environment object.</i>
-----------------	---

Description

R interface function for LINGO API function LScrtateEnvLng. For more information, please refer to LINGO User Manual.

Usage

```
rLScrtateEnvLng()
```

Value

If successful, rLScrtateEnvLng returns a LINGO environment object; Otherwise, it returns NULL.

References

LINDO SYSTEMS home page at www.lindo.com

See Also

[rLScrtateEnvLng](#) [rLScrtateEnvLicenseLng](#)

rLScrtateEnvLng	<i>Deletes a LINGO environment object previously created through a call to rLScrtateEnvLng.</i>
-----------------	---

Description

R interface function for LINGO API function LScrtateEnvLng. For more information, please refer to LINGO User Manual.

Usage

```
rLScrtateEnvLng(env)
```

Arguments

env	A LINGO environment object.
-----	-----------------------------

Value

An R list object with components:

ErrorCode Zero if successful, nonzero otherwise.

References

LINDO SYSTEMS home page at www.lindo.com

rLSEXecuteScriptLng	<i>Main workhorse of the LINGO DLL that processes LINGO command scripts.</i>
---------------------	--

Description

R interface function for LINGO API function LSEXecuteScriptLng. For more information, please refer to LINGO User Manual.

Usage

```
rLSEXecuteScriptLng(env, pcScript)
```

Arguments

env A LINGO environment object.

pcScript A character string containing a LINGO command script.

Value

An R list object with components:

ErrorCode Zero if successful, nonzero otherwise.

References

LINDO SYSTEMS home page at www.lindo.com

rLSopenLogFileLng	<i>Creates a file for LINGO to write a log to while processing your script.</i>
-------------------	---

Description

R interface function for LINGO API function LSopenLogFileLng. For more information, please refer to LINGO User Manual. When using this function, make sure Lingo have write permission to the folder where the Lingo log file will be sent.

Usage

```
rLSopenLogFileLng(env, pcLogFile)
```

Arguments

env	A LINGO environment object.
pcLogFile	A string containing the pathname for the log file.

Value

An R list object with components:

ErrorCode	Zero if successful, nonzero otherwise.
-----------	--

References

LINDO SYSTEMS home page at www.lindo.com

See Also

[rLScloseLogFileLng](#)

rLSparam	<i>LINGO API Parameters.</i>
----------	------------------------------

Description

For more information, please refer to LINGO User Manual.

References

LINDO SYSTEMS home page at www.lindo.com

rLSsetCharPointerLng	<i>Passes a list of character type memory pointers to LINGO.</i>
----------------------	--

Description

R interface function for LINGO API function LSsetCharPointerLng. For more information, please refer to LINGO User Manual.

Usage

```
rLSsetCharPointerLng(env, pacPointer, pnPointersNow)
```

Arguments

env	A LINGO environment object.
pacPointer	A string array whose memory location will be used by an instance of @POINTER().
pnPointersNow	A one dimensional integer array in which LINGO returns the current number of entries in the @POINTER() pointer list.

Value

An R list object with components:

ErrorCode Zero if successful, nonzero otherwise.

References

LINDO SYSTEMS home page at www.lindo.com

See Also

[rLSclearPointersLng](#)

rLSsetDouPointerLng *Passes a list of double type memory pointers to LINGO.*

Description

R interface function for LINGO API function LSsetDouPointerLng. For more information, please refer to LINGO User Manual.

Usage

```
rLSsetDouPointerLng(env,padPointer,pnPointersNow)
```

Arguments

env	A LINGO environment object.
padPointer	A double type array whose memory location will be used by an instance of @POINTER().
pnPointersNow	A one dimensional integer array in which LINGO returns the current number of entries in the @POINTER() pointer list.

Value

An R list object with components:

ErrorCode Zero if successful, nonzero otherwise.

References

LINDO SYSTEMS home page at www.lindo.com

See Also

[rLSclearPointersLng](#)

rLSsetIntPtrLng *Passes a list of integer type memory pointers to LINGO.*

Description

R interface function for LINGO API function LSsetIntPtrLng. For more information, please refer to LINGO User Manual.

Usage

```
rLSsetIntPtrLng(env, panPointer, pnPointersNow)
```

Arguments

env	A LINGO environment object.
panPointer	A integer type array whose memory location will be used by an instance of @POINTER().
pnPointersNow	A one dimensional integer array in which LINGO returns the current number of entries in the @POINTER() pointer list.

Value

An R list object with components:

ErrorCode	Zero if successful, nonzero otherwise.
-----------	--

References

LINDO SYSTEMS home page at www.lindo.com

See Also

[rLSclearPointersLng](#)

Index

LS_DINFO_MIP_BEST_OBJECTIVE_LNG
(rLSparam), [7](#)
LS_DINFO_MIP_BOUND_LNG (rLSparam), [7](#)
LS_DINFO_OBJECTIVE_LNG (rLSparam), [7](#)
LS_DINFO_SUMINF_LNG (rLSparam), [7](#)
LS_IINFO_BRANCHES_LNG (rLSparam), [7](#)
LS_IINFO_CONSTRAINTS_LNG (rLSparam), [7](#)
LS_IINFO_CONSTRAINTS_NONLINEAR_LNG
(rLSparam), [7](#)
LS_IINFO_ITERATIONS_LNG (rLSparam), [7](#)
LS_IINFO_NONZEROS_LNG (rLSparam), [7](#)
LS_IINFO_NONZEROS_NONLINEAR_LNG
(rLSparam), [7](#)
LS_IINFO_VARIABLES_INTEGER_LNG
(rLSparam), [7](#)
LS_IINFO_VARIABLES_LNG (rLSparam), [7](#)
LS_IINFO_VARIABLES_NONLINEAR_LNG
(rLSparam), [7](#)
LS_STATUS_CUTOFF_LNG (rLSparam), [7](#)
LS_STATUS_FEASIBLE_LNG (rLSparam), [7](#)
LS_STATUS_GLOBAL_LNG (rLSparam), [7](#)
LS_STATUS_INFEASIBLE_LNG (rLSparam), [7](#)
LS_STATUS_INFORUNB_LNG (rLSparam), [7](#)
LS_STATUS_LOCAL_INFEASIBLE_LNG
(rLSparam), [7](#)
LS_STATUS_LOCAL_LNG (rLSparam), [7](#)
LS_STATUS_NUMERIC_ERROR_LNG (rLSparam),
[7](#)
LS_STATUS_UNBOUNDED_LNG (rLSparam), [7](#)
LS_STATUS_UNDETERMINED_LNG (rLSparam), [7](#)
LSERR_INFO_NOT_AVAILABLE_LNG
(rLSparam), [7](#)
LSERR_INVALID_INPUT_LNG (rLSparam), [7](#)
LSERR_INVALID_LICENSE_KEY_LNG
(rLSparam), [7](#)
LSERR_INVALID_NULL_POINTER_LNG
(rLSparam), [7](#)
LSERR_INVALID_VARIABLE_NAME_LNG
(rLSparam), [7](#)
LSERR_JNI_CALLBACK_NOT_FOUND
(rLSparam), [7](#)
LSERR_NO_ERROR_LNG (rLSparam), [7](#)
LSERR_OUT_OF_MEMORY_LNG (rLSparam), [7](#)
LSERR_UNABLE_TO_COMPLETE_TASK_LNG
(rLSparam), [7](#)
LSERR_UNABLE_TO_OPEN_LOG_FILE_LNG
(rLSparam), [7](#)
rLingo, [2](#)
rLSclearPointersLng, [3](#), [8](#), [9](#)
rLScloseLogFileLng, [4](#), [7](#)
rLScreateEnvLicenseLng, [4](#), [5](#)
rLScreateEnvLng, [5](#), [5](#)
rLSdeleteEnvLng, [5](#), [5](#)
rLSexecuteScriptLng, [6](#)
rLSopenLogFileLng, [4](#), [6](#)
rLSparam, [7](#)
rLSsetCharPointerLng, [3](#), [7](#)
rLSsetDouPointerLng, [3](#), [8](#)
rLSsetIntPointerLng, [3](#), [9](#)