

Package ‘RNetLogo’

February 14, 2012

Title Provides an interface to the agent-based modelling platform NetLogo

Version 0.9.2

Date 2012-01-08

Author Jan C. Thiele

Maintainer Jan C. Thiele <jthiele@gwdg.de>

Description Interface to embed NetLogo into the R environment with headless (no GUI) and interactive GUI mode. Provides functions to load models, execute commands and to get values from reporters. Equivalent to the NetLogo Mathematica Link <http://ccl.northwestern.edu/netlogo/docs/mathematica.html>

Depends R (>= 2.12.1), rJava

Suggests igraph

Imports rJava (>= 0.6-3)

SystemRequirements Java (>= 5.0), NetLogo (>= 4.1)

License GPL-2

LazyLoad yes

Repository CRAN

Date/Publication 2012-01-09 07:41:46

R topics documented:

RNetLogo-package	2
NLCommand	3
NLDfToList	4
NLDoCommand	5
NLDoCommandWhile	6
NLDoReport	7

NLDoReportWhile	8
NLGetAgentSet	10
NLGetGraph	11
NLGetPatches	12
NLLoadModel	13
NLQuit	14
NLReport	15
NLSetPatches	16
NLSourceFromString	17
NLStart	18

Index	21
--------------	-----------

RNetLogo-package	<i>Provides an interface to the agent-based modelling platform NetLogo</i>
------------------	--

Description

Interface to embed NetLogo into the R environment with headless (no GUI) and interactive GUI mode. Provides functions to load models, execute commands and to get values from reporters. Mostly equivalent to NetLogo Mathematica Link <http://ccl.northwestern.edu/netlogo/docs/mathematica.html>.

Details

Package:	RNetLogo
Type:	Package
Version:	0.9
Date:	2011-07-13
License:	GNU GPL v2
LazyLoad:	yes

Start with the creation of a NetLogo instance by using `NLStart`. Then load a model with the function `NLLoadModel` and then use a command and/or reporter to do what you like.

Please note, that the package is not tested with Mac OS. If you made experiences with the package on Mac, please let me know.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

References

For NetLogo see <http://ccl.northwestern.edu/netlogo>. For R Extension for NetLogo see <http://netlogo-r-ext.berlios.de>. The RNetLogo package is an equivalent (and inspired by)

to the NetLogo Mathematica Link <http://ccl.northwestern.edu/netlogo/docs/mathematica.html>.

See Also

[NLStart](#), [NLLoadModel](#), [NLQuit](#), rJava package

Examples

```
## Not run:
library(RNetLogo)
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""))
NLCommand("setup")
NLDoCommand(10, "go")
burned <- NLReport("burned-trees")
print(burned)

## End(Not run)
```

NLCommand

Executes a command in the referenced NetLogo instance.

Description

NLCommand function is used to execute a NetLogo command (submitted as a string) in the submitted NetLogo instance.

Usage

```
NLCommand(..., nl.obj=NULL)
```

Arguments

... An undefined number of strings with the NetLogo command(s) to be executed. Vectors, lists and data.frames will be represented as NetLogo-Lists. To set a NetLogo-List you can write 'set mylist', c(1, 2, 3) if the current NetLogo model knows a list named mylist. Furthermore, you can execute multiple commands at once, e.g. 'setup', 'go'

nl.obj (optional) A variable holding a reference to a NetLogo instance created with [NLStart](#).

Details

The command could be everything which can be submitted from the NetLogo Command Center. A command has no return value! If you want to return a value from NetLogo use [NLReport](#) and friends.

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLDoCommand](#), [NLDoCommandWhile](#), [NLReport](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
NLCommand("create-turtles 10")

## End(Not run)
```

NLDfToList	<i>Transforms a data.frame into a NetLogo-List or multiple NetLogo-List (one for each column of the data.frame).</i>
------------	--

Description

NLDfToList function is used to push the values of a data.frame in NetLogo-Lists. The column names of the data.frame are used as names for the NetLogo-Lists (but the NetLogo-List must already exist in the current NetLogo model).

Usage

```
NLDfToList(in.data.frame, nl.obj=NULL)
```

Arguments

<code>in.data.frame</code>	The data.frame which should be used to fill the NetLogo-Lists (with the names of the columns of this data.frame) with the corresponding values of the column.
<code>nl.obj</code>	(optional) A variable holding a reference to a NetLogo instance created with NLStart .

Details

Remember: There have to be NetLogo-Lists in the NetLogo model with the names of the columns of the submitted data.frame.

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLDoCommand](#), [NLDoCommandWhile](#), [NLReport](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
df1 <- data.frame(x=c(1,2,3,4),y=c(1,2,3,4))
# the current NetLogo model must have lists with the names 'x' and 'y'
NLDFToList(df1)

## End(Not run)
```

NLDoCommand	<i>Repeated execution of a command in the referenced NetLogo instance for a defined number of repetitions.</i>
-------------	--

Description

NLDoCommand function is used to execute a NetLogo command (submitted as a string) in the submitted NetLogo instance more than one time. It works like [NLCommand](#).

Usage

```
NLDoCommand(iterations, ..., nl.obj=NULL)
```

Arguments

iterations	An interger defining the number of repetitions of the execution.
...	An undefined number of string(s) with the NetLogo command(s) to be executed. See NLCommand for details.
nl.obj	(optional) A variable holding a reference to a NetLogo instance created with NLStart .

Details

This function is used to execute a command for more than one time. It is usually used to call a procedure (e.g. "go") for a defined number of times.

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLCommand](#), [NLDoCommandWhile](#), [NLReport](#)

Examples

```
## Not run:
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""))
NLCommand("setup")
NLDoCommand(10, "go")

## End(Not run)
```

NLDoCommandWhile	<i>Repeated execution of a command in the referenced NetLogo instance while a reporter returns TRUE.</i>
------------------	--

Description

NLDoCommandWhile function is used to execute a NetLogo command (submitted as a string) in the submitted NetLogo instance more than one time. It works like [NLCommand](#) but will be repeated while the reporter returns TRUE.

Usage

```
NLDoCommandWhile(condition, ..., nl.obj=NULL)
```

Arguments

condition	A string with a NetLogo conditional reporter.
...	An undefined number of string(s) with the NetLogo command(s) to be executed. See NLCommand for details.
nl.obj	(optional) A variable holding a reference to a NetLogo instance created with NLStart .

Details

This function is used to execute a command for more than one time. It could be used to run a simulation (e.g. calling "go") while a variable is lower a boundary value.

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLCommand](#), [NLDoCommandWhile](#), [NLReport](#)

Examples

```
## Not run:
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""))
NLCommand("setup")
NLDoCommandWhile("burned-trees < 2200", "go")

## End(Not run)
```

NLDoReport	<i>Repeated execution of a command and a reporter in the referenced NetLogo instance for a defined number of repetitions.</i>
------------	---

Description

NLDoReport function is used to execute a NetLogo command (submitted as a string) in the NetLogo instance for more than one time and to execute the reporter after each iteration. It works like a combination of [NLReport](#) and [NLDoCommand](#).

Usage

```
NLDoReport(iterations, command, reporter, as.data.frame=FALSE,
           df.col.names=NULL, nl.obj=NULL)
```

Arguments

iterations	An interger defining the number of repetitions of the execution.
command	A string with the NetLogo command to be executed.
reporter	A string conataining a NetLogo reporter. (Or a vector/list containing multiple strings with different reporters - but the same effect can be reached via (list var1 var2 var3).)
as.data.frame	(optional) If TRUE the function will return a data.frame instead a list. Default is FALSE which returns a list.

- `df.col.names` (optional) If `as.data.frame=TRUE` you can define the names of the columns of the returned `data.frame` via this parameter. Input should be a vector containing the names as strings in the same order as the submitted reporters.
- `nl.obj` (optional) A variable holding a reference to a NetLogo instance created with [NLStart](#).

Details

This function is used to execute a command for more than one time and report a value or a number of values after each iteration. It is often used to call a procedure (e.g. "go") for a defined number of times and will save the value of a state variable in each simulation step.

Value

A list/nested list or `data.frame` with the value(s) of the reporter after each execution of the command.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLDoCommand](#), [NLReport](#), [NLDoReportWhile](#)

Examples

```
## Not run:
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""))
NLCommand("setup")
burned10 <- NLDoReport(10, "go", "burned-trees")
initburned10 <- NLDoReport(10, "go", c("initial-trees","burned-trees"),
  as.data.frame=TRUE, df.col.names=c("initial","burned"))

## End(Not run)
```

NLDoReportWhile	<i>Repeated execution of a command and a reporter in the referenced NetLogo instance while a conditional reporter returns TRUE.</i>
-----------------	---

Description

NLDoReportWhile function is used to execute a NetLogo command (submitted as a string) in the submitted NetLogo instance more than one time and to execute the reporter after each iteration. It works like [NLDoReport](#) but will be repeated while the conditional reporter returns TRUE.

NLGetAgentSet *Reports the values of the variables of the agent or agentset as a list*

Description

NLGetAgentSet is an easy-to-use way to access variables of an agent or an agentset. An agent is a turtle, breed, patch or link. An agentset is a collection of agents.

Usage

```
NLGetAgentSet(agent.var, agentset, as.data.frame=FALSE,
              df.col.names=NULL, nl.obj=NULL)
```

Arguments

agent.var	A string or vector/list of strings with the names of the agent/agentset variables.
agentset	A string specifying the agent or agentset to be queried.
as.data.frame	(optional) If TRUE (and agent.var is a list or vector) the function will return a data.frame instead a list. Default is FALSE which returns a list. Tip: If you want to get more than one agent variable (agent.var is a list or vector with length > 1) it's much more faster to ask for a data.frame instead of a list with nested lists for each agent.
df.col.names	(optional) If as.data.frame=TRUE you can define the names of the columns of the returned data.frame via this parameter. Input should be a vector containing the names as strings in the same order as the submitted reporters.
nl.obj	(optional) A variable holding a reference to a NetLogo instance created with NLStart .

Details

It's possible to use all the variables of an agent, which can be found in the inspect window. It isn't possible to get values from different types of agents (i.e. turtles, patches, links) with one call of NLGetAgentSet.

Value

Returns a list with the value(s) of the agent/agentset variable(s). One (nested) list for each agent.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLReport](#), [NLGetPatches](#), [NLGetGraph](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
NLCommand("create-turtles 10")
colors <- NLGetAgentSet(c("who", "xcor", "ycor", "color"),
  "turtles with [who < 5]")
# it's equivalent to (but unsorted):
# colors <- NLReport("[list who xcor ycor color] of turtles
# with [who < 5]")

## End(Not run)
```

NLGetGraph

Captures a network.

Description

NLGetGraph returns a graph object.

Usage

```
NLGetGraph(link.agentset="links", nl.obj=NULL)
```

Arguments

link.agentset (optional) A string defining the set of links/network. Default is "links", which are all links.

nl.obj (optional) A variable holding a reference to a NetLogo instance created with [NLStart](#).

Details

Save a link network in a graph object of package igraph for network analysis.

Value

Return a graph object of package igraph.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLGetAgentSet](#)

Examples

```
## Not run:
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <-
"/models/Sample Models/Networks/Preferential Attachment.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""))
NLCommand("setup")
NLDoCommand(4, "go")
graph1 <- NLGetGraph()
plot(graph1, layout=layout.kamada.kawai, vertex.label=V(graph1)$name,
      vertex.shape="rectangle", vertex.size=20, asp=FALSE)

## End(Not run)
```

NLGetPatches

Reports the values of the variables of the patches as a list

Description

NLGetPatches is an easy-to-use way to access variables of all patches (default) or of an subset of patches.

Usage

```
NLGetPatches(patch.var, patchset="patches", as.matrix=FALSE,
            as.data.frame=FALSE, df.col.names=NULL,
            nl.obj=NULL)
```

Arguments

patch.var	A string or vector/list of strings with the names of patch-variables to be reported.
patchset	(optional) A string defining which patches should be requested. By default, values of all patches will be returned.
as.matrix	(optional) If this variable is TRUE, the function will return the result as a matrix representing the NetLogo world. (Only available if you don't change the argument patchset, i.e. if you request the all patches/the whole world.)
as.data.frame	(optional) If TRUE (and patch.var is a list or vector) the function will return a data.frame instead a list. Default is FALSE which returns a list. Tip: If you want to get more than one patch variable (patch.var is a list or vector with length > 1) it's much more faster to ask for a data.frame instead of a list with nested lists for each patch.
df.col.names	(optional) If as.data.frame=TRUE you can define the names of the columns of the returned data.frame via this parameter. Input should be a vector containing the names as strings in the same order as the submitted reporters.
nl.obj	(optional) A variable holding a reference to a NetLogo instance created with NLStart .

Details

It's possible to use all the variables of a patch, which can be found in the inspect window.

Value

Returns a list with the values of the patches variable(s). One (nested) list for each patch.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLReport](#), [NLGetAgentSet](#), [NLGetGraph](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
allpatches <- NLGetPatches(c("pxcor", "pycor", "pcolor"))
subsetpatches <- NLGetPatches(c("pxcor", "pycor", "pcolor"),
                             "patches with [pxcor < 5]")
# it's equivalent to (but unsorted):
# subsetpatches <- NLReport("[list pxcor pycor pcolor]
#                               of patches with [pxcor < 5]")
# and
# subsetpatches <- NLGetAgentSet(c("pxcor", "pycor", "pcolor"),
#                               "patches with [pxcor < 5]")

## End(Not run)
```

NLLoadModel

Loads a model into the NetLogo instance.

Description

NLLoadModel loads a model (*.nlogo file) into the submitted NetLogo instance.

Usage

```
NLLoadModel(model.path, nl.obj=NULL)
```

Arguments

`model.path` A string containing the path to the model file (*.nlogo file).
`nl.obj` (optional) A variable holding a reference to a NetLogo instance created with [NLStart](#).

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLStart](#), [NLQuit](#)

Examples

```
## Not run:
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
absolute.model.path <- paste(nl.path,model.path,sep="")
NLLoadModel(absolute.model.path)

## End(Not run)
```

NLQuit

Quits a NetLogo instance.

Description

Quits the NetLogo workspace and closes the GUI window (if started with GUI).

Usage

```
NLQuit(nl.obj=NULL)
```

Arguments

`nl.obj` (optional) A variable holding a reference to a NetLogo instance created with [NLStart](#).

Value

No return value.

Warning

There is currently no way to kill a NetLogo instance with GUI completely. After executing `NLQuit` you can't run `NLStart` again. You have to quit your R session first and start a new one. The reason is, that NetLogo quits itself via `System.exit` (and has no functionality to quit all threads manually) but executing `System.exit` will terminate the whole JVM which will terminate also `rJava` and finally `R`. It can happen that some memory is not released although you have executed `NLQuit`. Therefore, it is a good idea to start a new R session whenever it is possible (e.g. if you are going to load a new model).

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLStart](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
NLQuit()

## End(Not run)
```

NLReport

Reports a value or list of values

Description

NLReport reports NetLogo data back to R.

Usage

```
NLReport(reporter, n1.obj=NULL)
```

Arguments

`reporter` A string containing a NetLogo reporter. (Or a vector of strings.)
`n1.obj` (optional) A variable holding a reference to a NetLogo instance created with [NLStart](#).

Details

Every reporter (= a command which will return a value), which can be called in the NetLogo Command Center, can be called with `NLReport`.

Value

A list or, if necessary, a nested list with the reported values.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLDoReport](#), [NLDoReportWhile](#), [NLGetPatches](#), [NLGetAgentSet](#)

Examples

```
## Not run:
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""))
NLCommand("setup")
NLDoCommandWhile("burned-trees < 2200", "go")
noburned <- NLReport("burned-trees")

## End(Not run)
```

NLSetPatches

Function to set a patch variable of all patches of the NetLogo world to the values of a matrix.

Description

NLSetPatches is an easy-to-use way to set the values of all patches (NetLogo world) to the values of a matrix.

Usage

```
NLSetPatches(patch.var, in.matrix, nl.obj=NULL)
```

Arguments

patch.var	The name of the patch variable which should be set to the values of the matrix.
in.matrix	A matrix which is a representation of the NetLogo world (= same dimension).
nl.obj	(optional) A variable holding a reference to a NetLogo instance created with NLStart .

Details

The matrix must have the same x- and y-dimension as the NetLogo world with indices beginning with (1,1). The upper-left cell (1,1) of the matrix represents the upper-left patch of the NetLogo world independent of the origin of the NetLogo world.

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLReport](#), [NLGetAgentSet](#), [NLGetGraph](#), [NLDfToList](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
m1 <- matrix(1:1089 , 33)
NLSetPatches(m1, "pcolor")

## End(Not run)
```

NLSourceFromString *Function to create/append model source code from R to NetLogo.*

Description

NLSourceFromString gives the opportunity to create/append a NetLogo model source code dynamically from R.

Usage

```
NLSourceFromString(..., append.model=TRUE, nl.obj=NULL)
```

Arguments

...	An undefined number of strings containing NetLogo model source code which should be printed into the procedures tab. Line brakes within a string can be obtained be typing \n.
append.model	(optional) Determines wheter existing code in the procedures tab (i.e. a loaded model) will be appended by the new code or will be replaced. By default, all existing code will be appended.
nl.obj	(optional) A variable holding a reference to an NetLogo instance created with NLStart .

Details

This function only works with NetLogo instances with GUI. It doesn't work in headless mode.

Value

No return value.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLReport](#), [NLGetAgentSet](#), [NLGetGraph](#), [NLDfToList](#)

Examples

```
## Not run:
NLStart("C:/Program Files/NetLogo 4.1.3")
setup <- "to setup\n ca\n crt 10\nend \n"
go <- "to go\n ask turtles [\n set xcor random-xcor\n
      set ycor random-ycor\n ]\nend \n"
reporter1 <- "to-report noturtles\n report count turtles\n end \n"
NLSourceFromString(setup,go,reporter1, append.model=FALSE)
NLCommand("setup")
NLCommand("go")
noturtles <- NLReport("noturtles")
print(noturtles)

## End(Not run)
```

NLStart

Creates an instance of NetLogo

Description

NLStart can be used to create a new instance of NetLogo in headless (without the Graphical User Interface) or GUI mode.

Usage

```
NLStart(nl.path, gui=TRUE, obj.name=NULL, nl.version=4, is3d=FALSE)
```

Arguments

nl.path	Path to your NetLogo installation (the folder where the NetLogo.jar is).
gui	(optional) A boolean value: if TRUE, NetLogo will be started with GUI (only one instance with GUI can be created currently!). FALSE will start NetLogo in headless mode.

<code>obj.name</code>	(optional) A string with the name of the variable which should be created and will contain the reference to the NetLogo instance. After the execution of <code>NLStart</code> is finished, there will be a variable with the given name in the global scope. This variable is needed for all further transactions and is always the last argument of the other functions of the <code>RNetLogo</code> package. If you don't specify this variable, an intern (in <code>.rnetlogo</code> environment) variable will be created and automatically used in all further functions.
<code>nl.version</code>	(optional) An interger value, that determines, if you want to start a NetLogo version 4.x or a NetLogo version 5.x. Don't try to start a NetLogo version 4.x with <code>nl.version=5</code> and vice versa. It is not possible to mix NetLogo versions in one R session. Please use different R sessions if you want to start <code>RNetLogo</code> with version 4 and version 5.
<code>is3d</code>	(optional) A boolean value: if <code>TRUE</code> , NetLogo 3D will be started. <code>FALSE</code> will start the conventionally 2D NetLogo.

Details

You can start multiple instances of NetLogo in headless mode and save each in another variable but it is not possible to start multiple instances in GUI mode. (It would result in a crash of R since there is no way to detach the Java Virtual Machine via `rJava`.)

Note for Linux and Mac OS X (Mac is untested) users: If you want to use the GUI mode, then it can be necessary to run `RNetLogo` in JGR application (see package `JGR`, <http://cran.r-project.org/web/packages/JGR/index.html>).

General Notes: 1. You should avoid to manually change the working directory of R, because NetLogo need to have the working directory pointed to its installation path. As the R working directory and the Java working directory depend on each other, changing the R working directory can result in unexpected behavior of NetLogo. There, you should use absolute paths for I/O processes in R instead of submitting `setwd(...)`. Note, that the `RNetLogo` package changes the working directory automatically when loading NetLogo and changes back to the former working directory when submitting `NLQuit`. 2. It is currently not possible to quit NetLogo completely (because NetLogo quits itself via `System.exit` but executing `System.exit` will terminate the whole JVM which will terminate also `rJava` and finally R.). It can happen that some memory is not released although you have executed `NLQuit`. Therefore, it is a good idea to start a new R session whenever it is possible (e.g. if you are going to load a new model).

Value

The function `implicit` returns an Java Object containing the reference to the Java controlling interface stored in a variable with the name given in 3rd function argument. This variable is needed for further transactions (like `NLLoadModel` or `NLCommand`).

Warning

It's not possible to run multiple instances of NetLogo in GUI mode! Closing NetLogo from the NetLogo Window is blocked, because it would quit the whole R process. To close the NetLogo call `NLQuit`. If you use the headless mode you should first load a model with `NLLoadModel` before executing other commands and reporters.

Author(s)

Jan C. Thiele <jthiele@gwdg.de>

See Also

[NLQuit](#)

Examples

```
## Not run:
library(RNetLogo)
nl.path <- "C:/Program Files/NetLogo 4.1.3"
NLStart(nl.path)
NLCommand("create-turtles 10")
noturtles <- NLReport("count turtles")
print(noturtles)

# create a second NetLogo instance in headless mode (= without GUI)
# with explicit name of stored object
NLStart(nl.path, FALSE, "nlheadless1")
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""), nlheadless1)
NLCommand("setup", nl.obj=nlheadless1)
burned1 <- NLDoReport(20, "go", c("ticks","burned-trees"),
                      as.data.frame=TRUE,df.col.names=c("tick","burned"),
                      nl.obj=nlheadless1)

print(burned1)

# create a third NetLogo instance in headless mode (= without GUI)
# with explicit name of stored object
NLStart(nl.path, FALSE, "nlheadless2")
model.path <- "/models/Sample Models/Earth Science/Fire.nlogo"
NLLoadModel(paste(nl.path,model.path,sep=""), nlheadless2)
NLCommand("setup", nl.obj=nlheadless2)
burned2 <- NLDoReport(10, "go", c("ticks","burned-trees"),
                      as_dataframe=TRUE,dfcolnames=c("tick","burned"),
                      nl.obj=nlheadless2)

print(burned2)

## End(Not run)
```

Index

- *Topic **NetLogo**
 - RNetLogo-package, 2
- *Topic **\textasciitildeNLCommand**
 - NLCommand, 3
- *Topic **\textasciitildeNLDFToList**
 - NLDFToList, 4
- *Topic **\textasciitildeNLDoCommand-While**
 - NLDoCommandWhile, 6
- *Topic **\textasciitildeNLDoCommand**
 - NLDoCommand, 5
- *Topic **\textasciitildeNLDoReportWhile**
 - NLDoReportWhile, 8
- *Topic **\textasciitildeNLDoReport**
 - NLDoReport, 7
- *Topic **\textasciitildeNLGetAgentSet**
 - NLGetAgentSet, 10
- *Topic **\textasciitildeNLGetGraph**
 - NLGetGraph, 11
- *Topic **\textasciitildeNLGetPatches**
 - NLGetPatches, 12
- *Topic **\textasciitildeNLLoadModel**
 - NLLoadModel, 13
- *Topic **\textasciitildeNLQuit**
 - NLQuit, 14
- *Topic **\textasciitildeNLReport**
 - NLReport, 15
- *Topic **\textasciitildeNLSetPatches**
 - NLSetPatches, 16
- *Topic **\textasciitildeNLSourceFromString**
 - NLSourceFromString, 17
- *Topic **\textasciitildeNLStart**
 - NLStart, 18
- *Topic **\textasciitildeRNetLogo**
 - NLCommand, 3
 - NLDFToList, 4
 - NLDoCommand, 5
 - NLDoCommandWhile, 6
 - NLDoReport, 7
 - NLDoReportWhile, 8
 - NLGetAgentSet, 10
 - NLGetGraph, 11
 - NLGetPatches, 12
 - NLLoadModel, 13
 - NLQuit, 14
 - NLReport, 15
 - NLSetPatches, 16
 - NLSourceFromString, 17
 - NLStart, 18
- *Topic **agent-based**
 - RNetLogo-package, 2
- *Topic **individual-based**
 - RNetLogo-package, 2
- NLCommand, 3, 5–7, 19
- NLDFToList, 4, 17, 18
- NLDoCommand, 4, 5, 5, 7, 8
- NLDoCommandWhile, 4, 5, 6, 6, 7, 9
- NLDoReport, 7, 8, 9, 16
- NLDoReportWhile, 8, 8, 16
- NLGetAgentSet, 10, 11, 13, 16–18
- NLGetGraph, 10, 11, 13, 17, 18
- NLGetPatches, 10, 12, 16
- NLLoadModel, 2, 3, 13, 19
- NLQuit, 3, 14, 14, 19, 20
- NLReport, 3–10, 13, 15, 17, 18
- NLSetPatches, 16
- NLSourceFromString, 17
- NLStart, 2–6, 8–17, 18
- RNetLogo (RNetLogo-package), 2
- RNetLogo-package, 2