

Package ‘froth’

July 22, 2025

Title Emulate a 'Forth' Programming Environment

Version 1.1.0

Description Emulates a 'Forth' programming environment with added features to interface between R and 'Forth'. Implements most of the functionality described in the original ``Starting Forth" textbook <<https://www.forth.com/starting-forth/>>.

Depends R (>= 4.3.0)

Imports methods

Suggests markdown, knitr

License GPL-3

ByteCompile true

Encoding UTF-8

NeedsCompilation yes

URL <https://www.ahl27.com/froth/>

BugReports <https://github.com/ahl27/froth/issues/new/choose>

VignetteBuilder knitr

Author Aidan Lakshman [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-9465-6785>>)

Maintainer Aidan Lakshman <ahl27@pitt.edu>

Repository CRAN

Date/Publication 2024-03-04 15:40:06 UTC

Contents

froth-dictionary	2
froth-parse-source	3
froth-reset	4
froth-RInterface	4
save-load-froth	6
Index	8

froth-dictionary	<i>List/Export Installed froth Words</i>
------------------	--

Description

Functions to inspect and save installed froth words.

Usage

```
froth.dictionary()  
writeFrothDictionary(file="", ...)
```

Arguments

file	file to write to, or "" for the console
...	additional arguments passed to cat

Details

`froth.dictionary` will list all installed words, grouped by their type (built-in, alias, user-defined). `writeFrothDictionary` allows users to export their function definitions. The default argument will print out user-defined definitions to the console. This output can be redirected to a file by changing the file argument.

Value

None. `froth.dictionary` lists all installed words using [message](#), and `writeFrothDictionary` either prints to the console or to a file.

Author(s)

Aidan Lakshman <ahl27@pitt.edu>

See Also

[saveFrothSession](#) [loadFrothSession](#)

Examples

```
## Show all words  
froth.dictionary()  
  
## Define a few new words  
froth.parse(": MAKE_THREE 1 2 + . ;")  
froth.parse(": MAKE_FIVE 2 3 + . ;")  
  
## print out definition  
writeFrothDictionary()
```

froth-parse-source	<i>Read/evaluate froth code from R</i>
--------------------	--

Description

Function to run froth code from R.

Usage

```
froth.parse(inputline)
froth.source(filepath)
```

Arguments

inputline	A string to parse with froth
filepath	Path to a file containing froth or FORTH code to parse with froth

Details

These functions run the froth interpreter on strings read in either as arguments (`froth.parse`) or from a file (`froth.source`). Both functions will run froth code without having to enter the REPL.

Value

Invisibly returns an integer status code, with 0 corresponding to normal execution.

Author(s)

Aidan Lakshman <ahl27@pitt.edu>

Examples

```
## Add two numbers
froth.parse("1 2 + .")

## source a function to print a ASCII table called 'rect'
tf <- tempfile()
defn <- ': RECT 256 0 DO I 16 MOD 0= IF CR THEN ." * " LOOP ;'
writeLines(defn, con=tf)
froth.source(tf)
froth.parse('rect')
```

froth-reset	<i>Reset the froth session</i>
-------------	--------------------------------

Description

Resets the froth session to defaults. This deletes any user-defined functions and variables, and clears the stack.

Usage

```
froth.reset()
```

Value

None; called to reset internal froth stacks.

Author(s)

Aidan Lakshman <ahl27@pitt.edu>

Examples

```
froth.RDefine("rnorm", rnorm, 3L)
froth.reset()
froth.parse("5 0 1 rnorm .s")
# fr> rnorm ?
```

froth-RInterface	<i>Interface with froth from R</i>
------------------	------------------------------------

Description

Methods to communicate with the froth environment without dropping into a REPL.

Usage

```
froth.RPush(object)
froth.RPop(nobj=1L)
froth.RDefine(name, fun, nargs)
```

Arguments

object	An R object to push to the froth stack
nobj	Number of objects to pop from the froth stack
name	Froth name for fun; see Examples
fun	An R function to define within froth
nargs	Number of arguments expected for fun

Details

These functions allow interaction with the froth stack from R. `froth.RPush` and `froth.RPop` allow push/pop operations on the froth stack. These operations are called from R, so pushing any R object is supported.

Some functions are easier to define using R than froth. `froth.RDefine` creates a froth function wrapper to call a specified R function, and then builds it into the froth environment. This makes using functions like `rnorm` within froth easier; see below for an illustrative example.

Functions defined with `froth.RDefine` expect their arguments to be popped directly off the froth stack, with the top of the stack corresponding to the last argument of the function.

Value

`froth.RPop` returns a list with the top `nobj` elements of the stack.

`froth.RPush` and `froth.RDefine` invisibly return an integer corresponding to the status of the operation. 0 indicates normal completion.

Note

Functions defined with `froth.RDefine` will not be saved using `saveFrothSession`.

Author(s)

Aidan Lakshman <ahl27@pitt.edu>

Examples

```
## Example of calling rnorm in froth

## rnorm expects 3 arguments: rnorm(n, mean, sd)
froth.RDefine(name='R_rnorm', fun=rnorm, nargs=3L)

## Now we can call rnorm from froth using the 'R_rnorm' word.
## Note that the arguments are expected on the stack
## such that the top of the stack is `sd`,
## the second is `mean`, and the third is `n`.

## n
froth.RPush(5)

## mean
froth.RPush(0.0)

## sd
froth.RPush(1.0)

## running the function
## note this will push the results back onto the stack
froth.parse("R_rnorm")

## we can get the result with froth.RPop
```

```
froth.RPop(5L)

## As a oneliner: (doesn't return the values)
froth.parse("5 0 1 R_rnorm .s")
```

save-load-froth	<i>Save/Load froth Sessions</i>
-----------------	---------------------------------

Description

Methods to preserve user-defined entries and variables.

Usage

```
saveFrothSession(file=NULL, ...)
loadFrothSession(file=NULL)
```

Arguments

file	Path to a file used for saving/loading
...	Additional arguments passed to saveRDS

Details

`saveFrothSession` saves current user-defined methods and variables within the Froth dictionary to the file specified. Built-in methods are loaded when the package is attached, so these aren't saved. Note that methods defined using `froth.RDefine` are currently not able to be saved.

`loadFrothSession` will restart the froth environment, which will erase any current user-defined methods and variables. It then loads the contents of the the file specified into the current Froth session.

Value

None. `loadFrothSession` will update internal froth stacks, and `saveFrothSession` will save to a file.

Author(s)

Aidan Lakshman <ahl27@pitt.edu>

Examples

```
tf <- tempfile()
froth.RDefine('rnorm', rnorm, 3L)
saveFrothSession(tf)
froth.reset()
froth.parse("5 0 1 rnorm .s")
# fr> rnorm ?
```

```
loadFrothSession(tf)
froth.parse("5 0 1 rnorm .s")
```

Index

cat, [2](#)

froth-dictionary, [2](#)
froth-parse-source, [3](#)
froth-reset, [4](#)
froth-RInterface, [4](#)
froth.dictionary (froth-dictionary), [2](#)
froth.parse (froth-parse-source), [3](#)
froth.RDefine, [6](#)
froth.RDefine (froth-RInterface), [4](#)
froth.reset (froth-reset), [4](#)
froth.RPop (froth-RInterface), [4](#)
froth.RPush (froth-RInterface), [4](#)
froth.source (froth-parse-source), [3](#)

loadFrothSession, [2](#)
loadFrothSession (save-load-froth), [6](#)

message, [2](#)

rnorm, [5](#)

save-load-froth, [6](#)
saveFrothSession, [2](#), [5](#)
saveFrothSession (save-load-froth), [6](#)
saveRDS, [6](#)

writeFrothDictionary
 (froth-dictionary), [2](#)