Package ‘flifo’

October 13, 2022

Type Package
Title Don't Get Stuck with Stacks in R
Version 0.1.5
Date 2018-07-31
Description Functions to create and manipulate
       FIFO (First In First Out), LIFO (Last In First Out), and NINO (Not In or Never Out)
       stacks in R.
License MIT + file LICENSE
LazyData TRUE
Depends R (>= 3.1.3)
Imports bazar, pryr
VignetteBuilder knitr
Suggests knitr, stats, testthat
URL https://github.com/paulponcet/flifo
BugReports https://github.com/paulponcet/flifo/issues
RoxygenNote 6.0.1
NeedsCompilation no
Author Paul Poncet [aut, cre]
Maintainer Paul Poncet <paulponcet@yahoo.fr>
Repository CRAN
Date/Publication 2018-07-31 21:50:03 UTC

R topics documented:

   flifo ......................................................... 2
   is.empty.stack ........................................... 2
   is.stack .................................................. 3
   max_length ............................................... 4
   pop ....................................................... 4
flifo provides functions to create and manipulate FIFO (First In First Out), LIFO (Last In First Out), and NINO (Not In or Never Out) stacks in R, most notably:

- `fifo`, `lifo`, and `nino` to create empty stacks;
- `push` to insert an object into a stack;
- `pop` to retrieve an object from a stack.

**is.empty.stack**

This method tests if a stack `x` is empty.

**Usage**

```r
## S3 method for class 'stack'
is.empty(x)
```

**Arguments**

- `x` A stack.

**Value**

A logical, TRUE if `x` is empty.

**See Also**

The generic function `is.empty` in package `bazar`.
Description

The `fifo`, `lifo`, and `nino` functions create 'First In First Out', 'Last In First Out', and 'Not In or Never Out' stacks, respectively.

Usage

```r
is.stack(x)

is.fifo(x)

is.lifo(x)

is.nino(x)
```

```r
## S3 method for class 'stack'
as.list(x, ...)

fifo(max_length = Inf, max_size = Inf)

lifo(max_length = Inf, max_size = Inf)

nino(max_length = Inf, max_size = Inf)
```

Arguments

- `x`: An object to be tested or coerced.
- `...`: Additional arguments.
- `max_length`: numeric. The maximum (infinite by default) number of objects the stack can contain.
- `max_size`: numeric. The maximum (infinite by default) size of the stack, in octets.

Value

- `is.xxx` functions return a logical.
- `fifo`, `lifo`, and `nino` return an empty FIFO, LIFO, or NINO stack.

See Also

- `push`, `pop`.  

max_length

Maximum length of a stack

Description

The function `max_length` returns the maximum number of objects a stack can contain; this number can be changed with `max_length<-.`

Usage

```
max_length(.stack)
max_length(x) <- value
```

Arguments

- `.stack`, `x` A stack.
- `value` numeric. The new maximum length of the stack.

Value

`max_length` returns a (possibly infinite) nonnegative numeric.

pop

Retrieve an object from a stack

Description

The `pop` function retrieves the first reachable object from `.stack`.

Usage

```
pop(.stack)
```

Arguments

- `.stack` A stack.

Details

The `pop` function is not pure. Side effect is that `.stack` is modified in the calling environment.

Value

The object retrieved. If `.stack` is empty, an error is thrown.
Print a stack.

print.stack

See Also

push.

Examples

(s <- lifo(max_length = 3)) # empty LIFO
(push(s, 0.3)) #
(push(s, data.frame(x=1:2, y=2:3)))
obj <- pop(s) # get the last element inserted

Description

The function print.stack prints the class of the stack x (FIFO, LIFO, or NINO) and displays its next reachable object.

Usage

## S3 method for class 'stack'
print(x, ...)

Arguments

x A stack.

... Additional arguments.

Value

The stack x is returned invisibly.

See Also

push, pop.
Description

The push function inserts an object into \texttt{.stack}.

Usage

\texttt{push(.stack, x)}

Arguments

\begin{itemize}
  \item \texttt{.stack} \hspace{1cm} A stack.
  \item \texttt{x} \hspace{1cm} An object to insert in \texttt{.stack}.
\end{itemize}

Details

The push function is not pure. Side effects (made on purpose) are:

\begin{itemize}
  \item \texttt{.stack} is modified in the calling environment;
  \item \texttt{x} is removed (deleted) if it exists in the calling environment.
\end{itemize}

Value

\texttt{NULL} is returned invisibly.

See Also

\texttt{pop}.

Examples

\begin{verbatim}
(s <- lifo(max_length = 3)) # empty LIFO
(push(s, 0.3)) #
(push(s, data.frame(x=1:2, y=2:3)))
obj <- pop(s) # get the last element inserted
\end{verbatim}
**size**

---

### Size of a stack

**Description**

The function `size` returns the size of a stack, in bytes. The function `max_size` returns the maximum number of objects a stack can contain; this number can be changed with `max_size<-`.

**Usage**

```
size(.stack)
max_size(.stack)
max_size(x) <- value
```

**Arguments**

- `.stack` A stack.
- `x` A stack.
- `value` numeric. The new maximum size of the stack.

**Value**

`size` always returns a nonnegative numeric. `max_size` returns a (possibly infinite) nonnegative numeric.
Index

as.list.stack (is.stack), 3
fifo, 2
fifo (is.stack), 3
flifo, 2
flifo-package (flifo), 2
is.empty, 2
is.empty.stack, 2
is.fifo (is.stack), 3
is.lifo (is.stack), 3
is.nino (is.stack), 3
is.stack, 3
lifo, 2
lifo (is.stack), 3
max_length, 4
max_length<- (max_length), 4
max.size (size), 7
max.size<- (size), 7
nino, 2
nino (is.stack), 3
pop, 2, 3, 4, 5, 6
print.stack, 5
push, 2, 3, 5, 6
size, 7