

# stack egg

---

Provides LIFO queue (stack) operations.  
Extension for Chicken Scheme  
Version 1.3

**Kon Lovett**

---

## Table of Contents

<b>1</b>	<b>About this egg .....</b>	<b>1</b>
1.1	Version history .....	1
1.2	Requirements .....	1
1.3	Usage .....	1
<b>2</b>	<b>Documentation .....</b>	<b>2</b>
<b>3</b>	<b>License .....</b>	<b>4</b>
	<b>Index .....</b>	<b>5</b>

# 1 About this egg

## 1.1 Version history

- 1.3 Exports
- 1.2 Added clear-stack
- 1.1 Doc fix [thanks to Toby Butzon], rename of stack-pop-1!, bug fix
- 1.0 Initial release

## 1.2 Requirements

This egg requires the following extensions:

`srfi-1`, `syntax-case`

## 1.3 Usage

Load this egg like so:

```
(require-extension stack)
```

## 2 Documentation

stack is a set of procedures and macros supporting LIFO queue operations. The stack is treated as an independent object, which is modified in place. In opposition to the common pattern of a variable being re-assinged a new stack representation after each mutating operation.

**make-stack** [macro]  
(make-stack [LIST])

Returns a new stack, with optional initial elements from LIST.

**clear-stack** [macro]  
(clear-stack STACK)

Make STACK empty.

**stack?** [macro]  
(stack? STACK)

Is STACK a stack?

**stack-empty?** [macro]  
(stack-empty? STACK)

Returns #t for an empty STACK, #f otherwise.

**stack-length** [macro]  
(stack-length STACK)

Returns the number of elements on the STACK.

**stack-depth** [macro]  
(stack-depth STACK)

Returns the index of the last element on the STACK, or -1 when stack empty.

**stack-peek** [macro]  
(stack-peek STACK [INDEX])

Returns the element in STACK at INDEX. Index must be  $0 \leq \& \leq$  (stack-depth), defaults to 0 (top of stack).

**stack-poke!** [macro]  
(stack-poke! STACK VALUE [INDEX])

Changes the STACK element at INDEX to VALUE. Returns the modified stack. The stack is modified in place. Index must be  $0 \leq \& \leq$  (stack-depth), defaults to 0 (top of stack)

**stack-push!** [macro]  
(stack-push! STACK VALUE ...)

Pushes VALUE ... onto the STACK. Returns the modified stack. The stack is modified in place.

**stack-pop\*!** [macro]

(stack-pop\*! STACK)

Removes the top element from the **STACK** and returns it. The stack is modified in place. Does not check for the stack-empty condition!

**stack-pop!** [procedure]

(stack-pop! STACK)

Removes the top element from the **STACK** and returns it. The stack is modified in place.

**stack-cut!** [procedure]

(stack-cut! STACK START-DEPTH [END-DEPTH])

Removes the **STACK** elements from **START-DEPTH** thru **END-DEPTH** and returns a list of the stack elements. The stack is modified in place. The start-depth must be  $0 \leq \& \leq (\text{stack-depth})$ . The end-depth must be  $\text{start-depth} \leq \& \leq (\text{stack-depth})$ , defaults to start-depth.

**list->stack** [macro]

(list->stack LIST)

Returns the **LIST** as a stack. The resulting stack may share memory with the list and the list should not be modified after this operation.

**stack->list** [macro]

(stack->list STACK)

Returns the **STACK** as a list, where the first element of the list is the top element of the stack. The resulting list may share memory with the stack object and should not be modified.

**stack-for-each** [macro]

(stack-for-each STACK PROCEDURE)

Applies the **PROCEDURE** to each element of the **STACK**, in order of top to bottom.

**print-stack** [macro]

(print-stack STACK)

Prints the elements of the **STACK** to the (current-output-port).

### 3 License

Copyright (c) 2005, Kon Lovett. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the Software), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ASIS, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# Index

## C

clear-stack ..... 2

## L

list->stack ..... 3

## M

make-stack ..... 2

## P

print-stack ..... 3

## S

stack->list ..... 3

stack-cut! ..... 3

stack-depth ..... 2

stack-empty? ..... 2

stack-for-each ..... 3

stack-length ..... 2

stack-peek ..... 2

stack-poke! ..... 2

stack-pop! ..... 3

stack-pop\*! ..... 3

stack-push! ..... 2

stack? ..... 2