

F-operator egg

Shift/Reset Control Operators
Extension for Chicken Scheme
Version 0.6

Kon Lovett

Table of Contents

1	About this egg	1
1.1	Version history	1
1.2	Requirements	1
2	Documentation	2
2.1	Shift/Reset - Dynamically scoped shift/reset	2
2.2	BShift/BReset - Statically scoped shift/reset	3
2.3	GShift/GReset - Generalized shift/reset	4
3	Issues	6
4	Examples.....	7
5	License	8
	Index	9

1 About this egg

1.1 Version history

0.6	Exports
0.5	Rename <code>*v</code> <code>*-values</code>
0.4	Rename <code>H</code> datatype
0.3	<code>%shift/%reset</code> , <code>%bshift/%breset</code> , bug fix <code>bshiftv</code>
0.2	<code>shiftv/resetv</code> , <code>bshiftv/bresetv</code>
0.1	Initial release

1.2 Requirements

This egg requires the following extensions:

`datatype`, `Chicken 2.310+`

2 Documentation

The static delimited continuation operators `shift` and `reset`.

2.1 Shift/Reset - Dynamically scoped shift/reset

`(require-extension shift-reset)`

A "stuck on control" situation, a 'shift' without an enclosing 'reset', is an error.

`%reset` [macro]
`(%reset EXPRESSION ...)`

Evaluate the body `EXPRESSION ...` with a delimited continuation. The body will contain one or more instances of `(%shift ...)`.

Any use of `(dynamic-wind ...)` within the dynamic scope of the partial continuation will be **ignored**!

`%shift` [macro]
`(%shift PC-TAG EXPRESSION)`

Within the scope of `EXPRESSION` `PC-TAG` is bound to the reified partial continuation delimited by the enclosing `(%reset ...)`. Provide a value to the partial continuation using the form `(PC-TAG <something>)`.

`reset` [macro]
`(reset EXPRESSION ...)`

Evaluate the body `EXPRESSION ...` with a delimited continuation. The body will contain one or more instances of `(shift ...)`.

`shift` [macro]
`(shift PC-TAG EXPRESSION)`

Within the scope of `EXPRESSION` `PC-TAG` is bound to the reified partial continuation delimited by the enclosing `(reset ...)`. Provide a value to the partial continuation using the form `(PC-TAG <something>)`.

`reset-values` [macro]
`(reset-values EXPRESSION ...)`

Multiple value return version of `(reset ...)`. The body will contain one or more instances of `(shift-values ...)`.

`shift-values` [macro]
`(shift-values PC-TAG EXPRESSION)`

Multiple value return version of `(shift ...)`. Provide a value to the partial continuation using the form `(PC-TAG <something> ...)`.

2.2 BShift/BReset - Statically scoped shift/reset

(require-extension bshift-breset)

Invalid delimited continuations, what RC-TAG below represents, and "stuck on control" will generate an error.

%breset [macro]

(%breset RC-TAG EXPRESSION ...)

Evaluate the body EXPRESSION ... with a delimited continuation named RC-TAG. The body will contain one or more instances of (%bshift RC-TAG ...).

Any use of (dynamic-wind ...) within the dynamic scope of the partial continuation will be **ignored!**

%bshift [macro]

(%bshift RC-TAG PC-TAG EXPRESSION)

Within the scope of EXPRESSION PC-TAG is bound to the reified partial continuation delimited by the enclosing (%breset RC-TAG ...). Provide a value to the partial continuation using the form (PC-TAG <something>).

breset [macro]

(breset RC-TAG EXPRESSION ...)

Evaluate the body EXPRESSION ... with a delimited continuation named RC-TAG. The body will contain one or more instances of (bshift RC-TAG ...).

bshift [macro]

(bshift RC-TAG PC-TAG EXPRESSION)

Within the scope of EXPRESSION PC-TAG is bound to the reified partial continuation delimited by the enclosing (breset RC-TAG ...). Provide a value to the partial continuation using the form (PC-TAG <something>).

breset-values [macro]

(breset-values RC-TAG EXPRESSION ...)

Multiple value return version of (breset ...). The body will contain one or more instances of (bshift-values RC-TAG ...).

bshift-values [macro]

(bshift-values RC-TAG PC-TAG EXPRESSION)

Multiple value return version of (bshift ...). Provide a value to the partial continuation using the form (PC-TAG <something> ...).

range [macro]

(range RC-TAG FROM VALUE STEP TO?)

The value of the delimited continuation ranges over the set of values specified by the state generation procedure suite. For use with (breset ...).

FROM

Zero argument procedure, returning the initial state.

VALUE

Single argument procedure, of the state, returning the value of the state.

STEP

Single argument procedure, of the state, returning the next state.

TO?Single argument procedure, of the state, returning **#t** when the range is complete.**range** [macro]

(range RC-TAG FROM [STEP] TO)

The value of the delimited continuation ranges over the number interval [FROM TO], by STEP. The increment is 1 when missing. For use with (breset ...).

%range [macro]

(%range RC-TAG FROM VALUE STEP TO?)

Version of (range ...) for use with (%breset ...).

%range [macro]

(%range RC-TAG FROM [STEP] TO)

Version of (range ...) for use with (%breset ...).

2.3 GShift/GReset - Generalized shift/reset

(require-extension gshift-greset)

The generalized shift and reset operator family from [How to remove a dynamic prompt: static and dynamic delimited continuation operators are equally expressible](#).**greset** [macro]

(greset HR E)

Reset parameterized by the H Reset procedure HR.

gshift [macro]

(gshift HS F E)

Shift parameterized by the H Shift procedure HS.

hr-stop [procedure]

(hr-stop V)

H Reset Stop.

hs-stop [procedure]

(hs-stop V)

H Shift Stop.

hr-prop [procedure]

(hr-prop V)

H Reset Propagate.

hs-prop	[procedure]
(hs-prop <i>V</i>)	
<i>H</i> Shift Propagate.	
H	[procedure]
(H <i>F</i> <i>X</i>)	
Returns the composition of <i>F</i> and <i>X</i> as an <i>H</i> -datatype.	
HV	[procedure]
(HV <i>V</i>)	
Returns the value of <i>V</i> as an <i>H</i> -datatype.	
H?	[procedure]
(H? <i>OBJECT</i>)	
Is <i>OBJECT</i> an <i>H</i> -datatype?	
case-H	[macro]
(case-H <i>E</i> ((<i>F</i> <i>X</i>) <i>ON-H-EXPR</i>) (<i>V</i> <i>ON-V-EXPR</i>))	
Deconstructs the <i>H</i> -datatype <i>E</i> , binding <i>F</i> & <i>X</i> for an evaluation of the <i>ON-H-EXPR</i> and <i>V</i> for an evaluation of the <i>ON-V-EXPR</i> .	

3 Issues

Not a direct implementation of partial continuations. Simulated using full continuations.

No multiple valued generalized shift/reset.

4 Examples

```
;;  
;; See "F-operator-test.scm" in the unpacked egg for more examples.  
;;
```

5 License

Copyright (c) 2006, 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.

Does not supercede any restrictions found in the source code.

Index

%

%breset	3
%bshift	3
%range	4
%reset	2
%shift	2

B

breset	3
breset-values	3
bshift	3
bshift-values	3

C

case-H	5
--------------	---

G

greset	4
--------------	---

gshift	4
--------------	---

H

H	5
H?	5
hr-prop	4
hr-stop	4
hs-prop	5
hs-stop	4
HV	5

R

range	3, 4
reset	2
reset-values	2

S

shift	2
shift-values	2