

Title

Syntax for dynamic scoping.

Author

Lars T Hansen

Status

This SRFI is currently in *withdrawn* status. Here is [an explanation](#) of each status that a SRFI can hold. To provide input on this SRFI, please send email to srfi-15@srfi.schemers.org. To subscribe to the list, follow [these instructions](#). You can access previous messages via the mailing list [archive](#).

- Received: 1999-11-01
- Draft: 1998-11-06--2000-01-07
- Withdrawn: 2000-03-10

Abstract

FLUID-LET, a binding syntax for dynamic scoping, is introduced.

Issues

None.

Rationale

FLUID-LET reduces the clutter of overriding the values of variables in a dynamic scope while allowing nonlocal exits from and re-entries to that scope. Scheme systems have had FLUID-LET for years, though they do not agree on its meaning.

Specification

(FLUID-LET <bindings> <body>)

Syntax

<Bindings> should have the form ((<variable1> <init1>) ...) where each <init> is an expression. It is an error for a <variable> to appear more than once in the list of variables.

The <init>s are evaluated in the current environment, the values of the <variable>s are saved, the results of the <init>s are assigned to the <variable>s, and the <body> is evaluated in the current environment before the <variable>s are restored to their saved values and the values returned by the last expression of <body> are returned as the values of the entire expression.

It is an error for the <variable>s to be unbound at the time their old values are saved.

If control leaves the dynamic scope of the <body> before the last expression of <body> has returned, then the current values of the <variables>s inside the scope are saved and the saved values from outside the scope are restored before control leaves the scope of the FLUID-LET. If control subsequently re-enters the scope of the <body>, then the current values of the <variable>s outside the scope are saved and the saved values from inside the scope are restored before execution continues inside <body>. In this case, it is the last saved values from outside the scope that will be restored when the <body> finally returns.

```
(define v 1)
(define again #f)

(define (test1)
  (display v)
  (fluid-let ((v 2))
    (call-with-current-continuation
     (lambda (k)
       (set! again (lambda ()
                     (set! again #f)
                     (k #t))))))

  (test2)
  (set! v 3))
(display v)
(set! v 4)
(if again (again)))
```

```
(define (test2) (display v))
```

```
(test1)
```

```
--> 12134
```

Implementation

The following implementation is written in R5RS Scheme. It is not compatible with the IEEE Scheme standard because the IEEE standard does not contain the high-level macro system.

The implementation assumes that some top-level names defined by the R5RS are bound to their original values.

```
(define-syntax fluid-let
  (syntax-rules ()
    ((_ ((v1 e1) ...) b1 b2 ...)
      (fluid-let "temps" () ((v1 e1) ...) b1 b2 ...))
    ((_ "temps" (t ...) ((v1 e1) x ...) b1 b2 ...)
      (let ((temp e1))
        (fluid-let "temps" ((temp e1 v1) t ...) (x ...) b1 b2 ...)))
    ((_ "temps" ((t e v) ...) () b1 b2 ...)
      (let-syntax ((swap!
                    (syntax-rules ()
                      ((swap! a b)
                       (let ((tmp a))
                         (set! a b)
                         (set! b tmp))))))
        (dynamic-wind
          (lambda ()
            (swap! t v) ...)
          (lambda ()
            b1 b2 ...)
          (lambda ()
            (swap! t v) ...))))))
```

Copyright

Copyright (C) Lars T Hansen (1999). 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 "AS IS", 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.

Editor: [Mike Sperber](#)

Last modified: Fri Sep 18 17:00:11 MST 2009