

File created: 17-May-90 16:11:33 {DSK}<usr>local>lde>lispcore>sources>UNWINDMACROS.;2

changes to: (VARS UNWINDMACROSCOMS)

previous date: 27-May-87 16:49:53 {DSK}<usr>local>lde>lispcore>sources>UNWINDMACROS.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

::  
:: Copyright (c) 1986, 1987, 1990 by Venue & Xerox Corporation. All rights reserved.

(RPAQQ **UNWINDMACROSCOMS**

(  
;; macros for use with the new unwinder

(FUNCTIONS NLSETQ ERSETQ)  
(MACROS RESETLST RESETFORM RESETVARS XNLSETQ RESETVAR RESETSAVE UNDONLSETQ)  
(PROP DMACRO CL:CATCH CL:THROW CL:UNWIND-PROTECT)  
(MACROS .CATCH. .UNWIND.PROTECT. .RESETLST.)  
(FNS COMP.CATCH COMP.UNWIND-PROTECT)  
(ADDVARS (SYSSPECVARS SI::\*DUMMY-FOR-CATCH\* SI::\*CATCH-RETURN-FROM\*))  
(PROP FILETYPE UNWINDMACROS)))

:: macros for use with the new unwinder

(DEFMACRO **NLSETQ** (&BODY FORMS)

;; Effectively (proceed-case (handler-bind ...)) but expanded by hand for efficiency.

`(LET (SI::NLSETQ-VALUE)  
 (CL:IF (EQ (LET ((\*PROCEED-CASES\* (CONS SI::NLSETQ-PROCEED-CASE \*PROCEED-CASES\*))  
 (SI::\*NLSETQFLAG\* T)  
 (\*CONDITION-HANDLER-BINDINGS\* (CONS '(CL:ERROR . SI::NLSETQHANDLER)  
 \*CONDITION-HANDLER-BINDINGS\*)))  
 (DECLARE (SPECVARS SI::\*NLSETQFLAG\*))  
 (CL:CATCH \*PROCEED-CASES\*  
 [CL:SETQ SI::NLSETQ-VALUE (LIST (PROGN ,@FORMS)  
 :NORMAL)])  
 :NORMAL)  
 SI::NLSETQ-VALUE  
 NIL)))

(DEFMACRO **ERSETQ** (&BODY FORMS)

;; Effectively (proceed-case ...), but hand-expanded for efficiency.

`(LET (SI::NLSETQ-VALUE)  
 (CL:IF (EQ (LET ((\*PROCEED-CASES\* (CONS SI::NLSETQ-PROCEED-CASE \*PROCEED-CASES\*))  
 (SI::\*NLSETQFLAG\* NIL))  
 (DECLARE (SPECVARS SI::\*NLSETQFLAG\*))  
 (CL:CATCH \*PROCEED-CASES\*  
 [CL:SETQ SI::NLSETQ-VALUE (LIST (PROGN ,@FORMS)  
 :NORMAL)])  
 :NORMAL)  
 SI::NLSETQ-VALUE  
 NIL)))

(DECLARE%: EVAL@COMPILE

(PUTPROPS **RESETLST MACRO** [(X . Y)  
 (.RESETLST. (PROGN X . Y)  
 NIL  
 ((LISPXHIST LISPXHIST)  
 (RESETSTATE NIL])

(PUTPROPS **RESETFORM MACRO** [TAIL `( .RESETLST. (PROGN ,@(CDR TAIL))  
 (LIST (LIST (LIST ',(CAAR TAIL)  
 ,(CAR TAIL])

(PUTPROPS **RESETVARS MACRO**

[TAIL (LET [(VARS (MAPCAR (CAR TAIL)  
 (FUNCTION (LAMBDA (Z)  
 (SETQ Z (MKLIST Z))  
 [COND  
 ([AND EMFLAG (NOT (COMP.GLOBALVARP (CAR Z)  
 (COMPERRM (LIST (CAR Z)  
 "- not GLOBALVAR in RESETVARS"]  
 Z])  
 `(.RESETLST. (PROG NIL ; Set the variables to new values, execute forms, all inside a prog  
 ,.[MAPCAR VARS (FUNCTION (LAMBDA (V)  
 (CONS 'SETQ V)  
 ,@(CDR TAIL)) ; Initialize \*RESETFORMS\* to list of vars and old values  
 (PROGN

```
(LIST ,@(MAPCAR VARS (FUNCTION (LAMBDA (V)
                                `(CONS ',(CAR V)
                                      ,(CAR V))
```

```
(PUTPROPS XNLSETQ MACRO ((X)
                          (NLSETQ X)))
```

```
(PUTPROPS RESETVAR MACRO [(VAR VAL FORM)
                            (.RESETLST. (PROGN (SETTOPVAL 'VAR VAL)
                                                FORM)
                                           (LIST (CONS 'VAR (GETTOPVAL 'VAR))
```

```
(PUTPROPS RESETSAVE MACRO
[X `(SETQ SI::*RESETFORMS*
  (CONS ,(COND
    [(AND (ATOM (CAR X))
          (CAR X))
     (SUBPAIR '(VAR VAL)
              X
              '(PROG1 (CONS 'VAR (GETTOPVAL 'VAR))
                      (SETTOPVAL 'VAR VAL)))]
    [(CDR X)
     `(LIST ,(CADR X)
            ,(CAR X)
            (T `(LIST (LIST ',(COND
                    ((EQ (CAAR X)
                        'SETQ)
                      (CAR (CADDAR X)))
                  (T (CAAR X)))
              ,(CAR X)
              SI::*RESETFORMS*))])
```

```
(PUTPROPS UNDONLSETQ MACRO ((UNDOFORM UNDOFN)
                              (PROG ((LISPXHIST LISPXHIST)
                                     UNDOSIDE0 UNDOSIDE UNDOTEM)
                                     (DECLARE (SPECVARS LISPXHIST))
                                     [COND
                                      ((LISTP (SETQ UNDOSIDE (LISTGET1 LISPXHIST 'SIDE)
                                                                    (SETQ UNDOSIDE0 (CDR UNDOSIDE))))
                                       (T (SETQ UNDOSIDE0 UNDOSIDE)
                                          (SETQ UNDOSIDE (LIST 0))
                                          (COND
                                           (LISPXHIST (LISTPUT1 LISPXHIST 'SIDE UNDOSIDE))
                                           (T (SETQ LISPXHIST (LIST 'SIDE UNDOSIDE))
                                              (RESETVARS (%#UNDOSAVES)
                                                         (SETQ UNDOTEM (XNLSETQ UNDOFORM NIL UNDOFN))))
                                          (COND
                                           ((EQ UNDOSIDE0 'NOSAVE)
                                            (LISTPUT1 LISPXHIST 'SIDE 'NOSAVE))
                                           (T (UNDOSAVE)))
                                          (COND
                                           (UNDOTEM (RETURN UNDOTEM)))
                                           (UNDONLSETQ1 (CDR UNDOSIDE)
                                                         (LISTP UNDOSIDE0))
                                           (RETURN))))))
```

```
(PUTPROPS CL:CATCH DMACRO ((TAG . BODY)
                              (.CATCH. TAG (PROGN . BODY))))
```

```
(PUTPROPS CL:THROW DMACRO (DEFMACRO (TAG VALUE) [COND
  [(NLISTP VALUE)
   ; simple one-valued case
   `(SI::INTERNAL-THROW ,TAG ,VALUE]
  [(EQ (CAR VALUE)
       'CL:VALUES)
   ; simple multi-valued case
   `(SI::INTERNAL-THROW ,TAG ,@(CDR VALUE)]
  (T
   ; general multi-valued case
   `(SI::INTERNAL-THROW-VALUES ,TAG (
                                     CL:MULTIPLE-VALUE-LIST
                                     ,VALUE]))
```

```
(PUTPROPS CL:UNWIND-PROTECT DMACRO (DEFMACRO (FORM &REST CLEANUP-FORMS) `(CL:MULTIPLE-VALUE-PROG1
  (.UNWIND.PROTECT.
   [FUNCTION ,(COND
    ((AND
     (NULL (CDR
            CLEANUP-FORMS)
            ))
    (LISTP
     (CAR
      CLEANUP-FORMS)
     ))
    (NULL (CDAR
           CLEANUP-FORMS)
           )))
```

```

; Optimize case of no-argument single cleanup fn
(CAAR CLEANUP-FORMS
)
(T
  (LAMBDA NIL
    ,@CLEANUP-FORMS]
,FORM)
,@CLEANUP-FORMS))

```

```
(DECLARE%: EVAL@COMPILE
```

```
(PUTPROPS .CATCH. DMACRO (APPLY COMP.CATCH))
```

```
(PUTPROPS .UNWIND.PROTECT. DMACRO (APPLY COMP.UNWIND-PROTECT))
```

```
(PUTPROPS .RESETLST. DMACRO (DEFMACRO (FORM &OPTIONAL INIT OTHERBINDINGS) `(LET
  ((SI::*RESETFORMS* ,INIT)
  ,@OTHERBINDINGS)
  [DECLARE
  (SPECVARS
  SI::*RESETFORMS*
  ,@(MAPCAR OTHERBINDINGS
  'CAR]
  (CL:UNWIND-PROTECT
  ,FORM
  (SI::RESETUNWIND))))))
)
```

```
(DEFINEQ
```

**(COMP.CATCH**

```
[LAMBDA (ARG FORM)
```

; Edited 27-May-87 16:48 by bvm:

;;; Compiles the separate subfunction for CATCH. The sub-function is a fn of one argument, ARG (the catch tag). FORM is the code to execute inside  
 the subfn

;; SI::\*DUMMY-FOR-CATCH\* atrocity is to get the catch tag in pvar 1--assumes bytecompiler does not gratuitously rearrange pvars. Avoid naming  
 ;; ivar0 to reduce clutter of name table.

```
(COMP.CALL [COMP.LAM1 `(LAMBDA NOBIND
  (LET [(SI::*DUMMY-FOR-CATCH* T)
  (SI::*CATCH-RETURN-FROM* ((OPCODES (IVAR 0)
  (DECLARE (CL:SPECIAL SI::*DUMMY-FOR-CATCH* SI::*CATCH-RETURN-FROM*))
  ,FORM]
  (LIST ARG)
  0])
)
```

**(COMP.UNWIND-PROTECT**

```
[LAMBDA (CLEANUPFN FORM)
```

(\* bvm%: " 1-Jul-86 11:42")

;;; Compiles the separate subfunction for UNWIND-PROTECT and friends. Frame's name is SI::\*UNWIND-PROTECT\* and its sole arg is the cleanup  
 fn. FORM is the form to execute inside the separate frame.

```
(COMP.CALL [COMP.LAM1 `(LAMBDA (SI::*CLEANUP-FORMS*)
  (DECLARE (SPECVARS SI::*CLEANUP-FORMS*))
  (\CALLME 'SI::*UNWIND-PROTECT*)
  ,FORM]
  (LIST CLEANUPFN)
  0])
)
```

```
(ADDTOVAR SYSSPECVARS SI::*DUMMY-FOR-CATCH* SI::*CATCH-RETURN-FROM*)
```

```
(PUTPROPS UNWINDMACROS FILETYPE COMPILE-FILE)
```

```
(PUTPROPS UNWINDMACROS COPYRIGHT ("Venue & Xerox Corporation" 1986 1987 1990))
```

---

**FUNCTION INDEX**

COMP.CATCH .....3      COMP.UNWIND-PROTECT .....3

---

**MACRO INDEX**

.CATCH. ....3	CL:CATCH .....2	RESETFORM .....1	RESETVAR .....2	UNDONLSETQ .....2
.RESETLST. ....3	ERSETQ .....1	RESETLST .....1	RESETVARS .....1	CL:UNWIND-PROTECT .2
.UNWIND.PROTECT. .3	NLSETQ .....1	RESETSAVE .....2	CL:THROW .....2	XNLSETQ .....2

---

**PROPERTY INDEX**

UNWINDMACROS .....3

---

**VARIABLE INDEX**

SYSSPECVARS .....3

---