File created: 13-Apr-88 15:12:02 **{ERIS}<VANMELLE>LISP>DEFINERPRINT.;20**

changes to: (FNS PRINDOTP SUPERPRINT/COMMENT PRINTDEF1)
(VARS DEFINERPRINTCOMS)
(PROPS (DEFINE-SPECIAL-FORM :DEFINITION-PRINT-TEMPLATE))
(FUNCTIONS XCL::PPRINT-DEFINER)

previous date: 13-Apr-88 13:04:04 {ERIS}<VANMELLE>LISP>DEFINERPRINT.;19

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
;; Copyright (c) 1988 by Xerox Corporation.  All rights reserved.

(RPAQQ DEFINERPRINTCOMS
       ((FUNCTIONS XCL::PPRINT-DEFINER XCL::PPRINT-DEFINER-FITP XCL::PPRINT-DEFINER-RECURSE)
        (PROP :DEFINITION-PRINT-TEMPLATE DEFCOMMAND CL:DEFCONSTANT DEFDEFINER DEFGLOBALPARAMETER DEFGLOBALVAR
              DEFINE-CONDITION CL:DEFINE-MODIFY-MACRO CL:DEFINE-SETF-METHOD DEFINE-SPECIAL-FORM DEFINLINE
              DEFMACRO CL:DEFPARAMETER CL:DEFSETF CL:DEFSTRUCT CL:DEFTYPE CL:DEFUN CL:DEFVAR)
        (COMS                                                   ; Macros for some things pp handles stupidly
              (FNS CODEWRAPPER.PRETTYPRINT PROG1.PRETTYPRINT CASE.PRETTYPRINT PROGV.PRETTYPRINT
                   INDENTATION.FROM.HERE SEQUENTIAL.PRETTYPRINT)
              (ALISTS (PRETTYPRINTMACROS UNINTERRUPTABLY CL:UNWIND-PROTECT RESETLST CL:BLOCK CL:IF PROG1 CL:WHEN
                             CL:UNLESS WITH-READER-ENVIRONMENT CL:CATCH CASE CL:ECASE CL:ETYPECASE CL:TYPECASE
                             CL:PROGV WITH.MONITOR)
                      (PRETTYEQUIVLST PROG* CL:COMPILER-LET)))
        [COMS                                                   ; Repairs to other prettyprinting functions
              (FNS SUPERPRINT/COMMENT PRIN2-LONG-STRING SUPERPRINT/WRAPPER SUPERPRINT/SPACE PRINENDLINE PRINDOTP
                   PRINTDEF1)
              (ADVISE MAKEFILE)
              (DECLARE%: EVAL@COMPILE DOCOPY                    ; Doing this at compile suppresses dwim junk
                     (P (MOVD? '\DSPRETTY/ENDLINE 'SUBPRINT/ENDLINE NIL T)))
              (DECLARE%: EVAL@COMPILE DONTCOPY [P (CL:PROCLAIM '(CL:SPECIAL **COMMENT**FLG
                                                                 *PRINT-SEMICOLON-COMMENTS* COMMENTFONT
                                                                 FNSLST RMARGIN SPACEWIDTH]
                     (FILES (LOADCOMP)
                            DSPRINTDEF))
              (DECLARE%: DONTEVAL@LOAD DOCOPY                   ; Backward compatibility, needed in Lyric especially
                     (P (MOVD 'XCL::PPRINT-DEFINER 'PPRINT-DEFINER NIL T]
        (PROP (FILETYPE MAKEFILE-ENVIRONMENT)
              DEFINERPRINT)))


(CL:DEFUN XCL::PPRINT-DEFINER (XCL::DEFINE-EXPRESSION)
   (DECLARE (CL:SPECIAL FORMFLG SPACEWIDTH))                    ; Bound in prettyprinter
   (COND
      ((OR (NULL FORMFLG)
           (CL:ATOM (CDR XCL::DEFINE-EXPRESSION)))              ; Degenerate cases or printing as a quoted form--punt to default
                                                                ; prettyprinting
       XCL::DEFINE-EXPRESSION)
      (T (LET ((TAIL XCL::DEFINE-EXPRESSION)
               (LEFT (DSPXPOSITION))
               XCL::TEMPLATE XCL::TOP-LEVEL-P XCL::NEXT TYPE XCL::FORM XCL::NEWLINEP)
              (DECLARE (CL:SPECIAL TAIL LEFT))                  ; For comment printer
              (CL:SETQ XCL::TOP-LEVEL-P (EQ LEFT (DSPLEFTMARGIN)))
                                                                ; Printing definition to file, etc.
              (CL:SETQ LEFT (+ LEFT (CL:* 3 SPACEWIDTH)))       ; Place we will indent body
              (PRIN1 "(")
              (PRIN2 (CAR TAIL))
              [CL:SETQ XCL::TEMPLATE (OR (GET (CL:POP TAIL)
                                              :DEFINITION-PRINT-TEMPLATE)
                                         '(:NAME]

              ;; This code should, and doesn't, pay attention to the NAME function to determine where the name is to decide what should and
              ;; shouldn't be bold. Right now, it always bolds the second thing. Fortunately, we currently don't have any definers that don't have
              ;; either the second or CAR of the second as the definition name.

              ;; Also, this code should be careful about calling the NAME function on the form.  Sometimes, the form is not really a call to the
              ;; definer but instead a back-quoted expression in a macro.  In most such cases, the name is not really there; some comma-quoted
              ;; expression is there instead.

              [WHILE (CL:CONSP TAIL)
                 DO (COND
                        ((AND (CL:LISTP (CL:SETQ XCL::NEXT (CAR TAIL)))
                              (EQ (CAR XCL::NEXT)
                                  COMMENTFLG)
                              (SEMI-COLON-COMMENT-P XCL::NEXT))  ; Comments can appear anywhere, so print this one without
                                                                 ; consuming the template.  ENDLINE has side effect of printing
                                                                 ; comments
                         (SUBPRINT/ENDLINE LEFT *STANDARD-OUTPUT*)
                         (CL:SETQ XCL::NEWLINEP T))
                        ((OR (CL:ATOM XCL::TEMPLATE)
                             (EQ (CL:SETQ TYPE (CL:POP XCL::TEMPLATE))
                                 :BODY))                         ; Once we hit the body, there's nothing more special to do.
```

```
                        (RETURN))
                       (T (SPACES 1)
                          (CASE TYPE
                              (:NAME                                ; Embolden the name of this thing
                                 (CL:SETQ XCL::NEWLINEP NIL)
                                 [COND
                                    ((NOT XCL::TOP-LEVEL-P)         ; Nothing special here--could even be a backquoted thing
                                     (XCL::PPRINT-DEFINER-RECURSE))
                                    (T (CL:POP TAIL)
                                       (COND
                                          ((CL:CONSP XCL::NEXT)     ; Name is a list.  Assume the real name is the car and the rest is
                                                                    ; an options list or something
                                           (CL:UNLESS (EQ (DSPYPOSITION)
                                                          (PROGN (PRIN1 "(")
                                                                 (PRINTOUT NIL .FONT LAMBDAFONT .P2
                                                                        (CAR XCL::NEXT)
                                                                        .FONT DEFAULTFONT)
                                                                 (SPACES 1)
                                                                 (PRINTDEF (CDR XCL::NEXT)
                                                                        T T T FNSLST)
                                                                 (PRIN1 ")")
                                                                 (DSPYPOSITION)))
                                                                    ; This thing took more than one line to print, so go to new line
                                               (SUBPRINT/ENDLINE LEFT *STANDARD-OUTPUT*)
                                               (CL:SETQ XCL::NEWLINEP T)))
                                          (T                        ; Atomic name is bold
                                             (PRINTOUT NIL .FONT LAMBDAFONT .P2 XCL::NEXT .FONT DEFAULTFONT])
                              (:ARG-LIST                            ; NEXT is some sort of argument list.
                                 (COND
                                    ((NULL XCL::NEXT)               ; If NIL, be sure to print as ()
                                     (PRIN1 "()")
                                     (CL:POP TAIL))
                                    (T (XCL::PPRINT-DEFINER-RECURSE)))
                                 (CL:SETQ XCL::NEWLINEP NIL))
                              (T                                    ; Just print it, perhaps starting a new line
                                 (CL:UNLESS (OR XCL::NEWLINEP (XCL::PPRINT-DEFINER-FITP XCL::NEXT))
                                                                    ; Go to new line if getting crowded
                                     (PRINENDLINE LEFT))
                                 (XCL::PPRINT-DEFINER-RECURSE)
                                 (CL:SETQ XCL::NEWLINEP NIL)))])

;; We've now gotten to the end of stuff we know how to print.  Just prettyprint the rest

(CL:UNLESS (NULL TAIL)
    (COND
       (XCL::NEWLINEP                                               ; Already on new line
          )
       ([OR (EQ TYPE :BODY)
            (NOT (XCL::PPRINT-DEFINER-FITP (CAR TAIL]
                                                                    ; Go to new line and indent a bit.  Always do this for the part
                                                                    ; matching &BODY, whether or not the prettyprinter thought that
                                                                    ; the remainder would "fit"
        (PRINENDLINE LEFT NIL T))
       (T (SPACES 1)))
    (WHILE [AND (CL:CONSP TAIL)
                (CL:ATOM (CL:SETQ XCL::FORM (CAR TAIL]
       DO
          ;; Print this doc string or whatever on its own line.  This is because otherwise the prettyprinter gets confused and tries
          ;; to put the next thing after the string
          (XCL::PPRINT-DEFINER-RECURSE)
          (CL:WHEN (AND (CL:KEYWORDP XCL::FORM)
                        (CL:CONSP TAIL))                            ; Some sort of keyword-value pair stuff--print it on same line
              (SPACES 1)
              (XCL::PPRINT-DEFINER-RECURSE))
          (CL:WHEN (NULL TAIL)
                (RETURN))
          (SUBPRINT/ENDLINE LEFT *STANDARD-OUTPUT*))
    (PRINTDEF TAIL T T T FNSLST))
 (PRIN1 ")")
 NIL))))
```

(CL:DEFUN **XCL::PPRINT-DEFINER-FITP** (XCL::ITEM)

;; True if it won't look silly to try to print ITEM at current position instead of starting new line

```
(CL:IF (CL:CONSP XCL::ITEM)
    (OR (EQ (CAR XCL::ITEM)
            COMMENTFLG)
        (AND (< (COUNT XCL::ITEM)
                20)
             (FITP XCL::ITEM)))
    (< (+ (DSPXPOSITION)
          (STRINGWIDTH XCL::ITEM *STANDARD-OUTPUT*))
       (DSPRIGHTMARGIN))))
```

(CL:DEFUN **XCL::PPRINT-DEFINER-RECURSE** ()

;; Print and pop the next element. Prettyprinter uses the variable IL:TAIL for lookahead

```
   (DECLARE (CL:SPECIAL TAIL))
   (SUPERPRINT (CAR TAIL)
         TAIL NIL *STANDARD-OUTPUT*)
   (CL:SETQ TAIL (CDR TAIL)))

(PUTPROPS DEFCOMMAND :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS CL:DEFCONSTANT :DEFINITION-PRINT-TEMPLATE (:NAME :VALUE))

(PUTPROPS DEFDEFINER :DEFINITION-PRINT-TEMPLATE (:NAME :TYPE :ARG-LIST :BODY))

(PUTPROPS DEFGLOBALPARAMETER :DEFINITION-PRINT-TEMPLATE (:NAME :VALUE))

(PUTPROPS DEFGLOBALVAR :DEFINITION-PRINT-TEMPLATE (:NAME :VALUE))

(PUTPROPS DEFINE-CONDITION :DEFINITION-PRINT-TEMPLATE (:NAME :VALUE :BODY))

(PUTPROPS CL:DEFINE-MODIFY-MACRO :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST))

(PUTPROPS CL:DEFINE-SETF-METHOD :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS DEFINE-SPECIAL-FORM :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS DEFINLINE :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS DEFMACRO :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS CL:DEFPARAMETER :DEFINITION-PRINT-TEMPLATE (:NAME :VALUE))

(PUTPROPS CL:DEFSETF :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :ARG-LIST :BODY))

(PUTPROPS CL:DEFSTRUCT :DEFINITION-PRINT-TEMPLATE (:NAME :BODY))

(PUTPROPS CL:DEFTYPE :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS CL:DEFUN :DEFINITION-PRINT-TEMPLATE (:NAME :ARG-LIST :BODY))

(PUTPROPS CL:DEFVAR :DEFINITION-PRINT-TEMPLATE (:NAME :VALUE))

;; Macros for some things pp handles stupidly

(DEFINEQ
```

(CODEWRAPPER.PRETTYPRINT

```
  [LAMBDA (FORM)                                          ; Edited 30-Mar-88 11:44 by bvm
    ;; Prettyprints things that wrap code like PROGN.  We usually want them to start the code on the next line, rather than put the first expression way
    ;; to the right of all the rest.
    (PRIN1 "(")
    (LET ((HERE (INDENTATION.FROM.HERE)))
         (PRIN2 (pop FORM))                               ; Print the "function" itself
         (if (NLISTP FORM)
             then                                         ; Ignore degenerate cases
                   (PRINTDEF FORM T T T FNSLST)
           else (SEQUENTIAL.PRETTYPRINT FORM HERE))
         (PRIN1 ")")
         NIL])
```

(PROG1.PRETTYPRINT

```
  [LAMBDA (EXPR)                                          ; Edited 30-Mar-88 12:02 by bvm
    ;; Prettyprinter advice for PROG1, CL:IF, UNLESS, etc.  Default way's main problem is that if the first expression is a non-list but some later
    ;; expression is a list, it doesn't put ALL the subsequent expressions equally indented.  Thus, you get something like (PROG1 A (expression) <cr>
    ;; (expression) ...)
    (if [OR [NLISTP (CDR (LISTP (CDR EXPR]
            (AND (NLISTP (CDDDR EXPR))
                 (for E in (LISTP (CADDR EXPR)) never (LISTP E]
        then                                              ; 2 or fewer elements, or 3 elements, the last of which is very
                                                          ; simple--let default prettyprinter do it
             EXPR
      else (PRIN1 "(")
           (LET [(HERE (INDENTATION.FROM.HERE))
                 (LEFT (PROGN (PRIN2 (pop EXPR))          ; Print the car of form
                              (SPACES 1)
                              (DSPXPOSITION]
                (DECLARE (SPECVARS LEFT))
                (if (OR (if (>= HERE LEFT)
                            then                          ; Default indentation wants to be greater than the function length,
                                                          ; so change it to here
                                (SETQ HERE LEFT))
                        (NLISTP (CAR EXPR))
                        (FITP (CAR EXPR)
                              NIL NIL NIL *STANDARD-OUTPUT*))
                    then (SUPERPRINT (CAR EXPR)
                                     EXPR NIL *STANDARD-OUTPUT*)   ; Print the first element right at this position
```

```
                    (pop EXPR))
               (SEQUENTIAL.PRETTYPRINT EXPR HERE))
          (PRIN1 ")")                                    ; Return NIL to say we handled it
          NIL])
```

(CASE.PRETTYPRINT

```
[LAMBDA (EXPR)                                           ; Edited 30-Mar-88 16:54 by bvm
  (if (NLISTP (CDR EXPR))
      then                                               ; Degenerate case--punt
          EXPR
    else
     (PRIN1 "(")
     (LET
      ((HERE (INDENTATION.FROM.HERE))
       (LEFT (PROGN (PRIN2 (pop EXPR))                   ; Print the car of form
                    (SPACES 1)
                    (DSPXPOSITION)))
       (TAIL EXPR)
       INNERLEFT CASE)
      (DECLARE (SPECVARS LEFT TAIL))
      (if (OR (if (>= HERE LEFT)
                  then                                   ; Default indentation wants to be greater than the function length,
                                                         ; so change it to here
                      (SETQ HERE LEFT))
              (NLISTP (CAR TAIL))
              (FITP (CAR TAIL)
                    NIL NIL NIL *STANDARD-OUTPUT*))
          then (SUPERPRINT (CAR TAIL)
                           TAIL NIL *STANDARD-OUTPUT*)   ; Print the first element right at this position
               (pop TAIL))
      [SETQ INNERLEFT (+ (SETQ LEFT HERE)
                         (TIMES 3 (CHARWIDTH (CHARCODE X)
                                             *STANDARD-OUTPUT*]
      (do
       (if (NLISTP TAIL)
           then (if TAIL
                    then                                 ; dotted tail?
                         (PRINENDLINE LEFT *STANDARD-OUTPUT*)
                         (PRINTDEF TAIL T T T))
                (PRIN1 ")")
                (RETURN NIL)
         elseif (SEMI-COLON-COMMENT-P (LISTP (CAR TAIL)))
           then                                          ; Print any comments stuck in between elements
                (SUPERPRINT/COMMENT (CAR TAIL)
                                    *STANDARD-OUTPUT*)
                (pop TAIL)
         else                                            ; Start new line, after printing any comments
         (PRINENDLINE LEFT *STANDARD-OUTPUT*)
         (if (NLISTP (SETQ CASE (CAR TAIL)))
             then                                        ; degenerate case?
                  (PRIN2 CASE)
           else (PRIN1 "(")
                (LET (FORMFLG)
                     (DECLARE (SPECVARS FORMFLG))        ; Print the key(s) as data
                     (SUPERPRINT (CAR CASE)
                                 CASE NIL *STANDARD-OUTPUT*)
                     (SPACES 1))
                [if (NLISTP (SETQ CASE (CDR CASE)))
                    then                                 ; No tail, but handle degenerates
                         (PRINTDEF CASE T T T)
                  else (SEQUENTIAL.PRETTYPRINT
                        CASE
                        (LET ((HERE (DSPXPOSITION)))
                             (if [OR (<= HERE INNERLEFT)
                                     (AND (NULL (CDR CASE))
                                          (if (LISTP (CDR CASE))
                                              then       ; Multiple things to print
                                                   NIL
                                            elseif (NLISTP (CAR CASE))
                                              then       ; Print simple consequent if space
                                                   (< (STRINGWIDTH (CAR CASE)
                                                                   *STANDARD-OUTPUT* T)
                                                      (- (DSPRIGHTMARGIN)
                                                         HERE))
                                              else (FITP CASE T NIL NIL *STANDARD-OUTPUT*]
                                 then                    ; Key didn't go too far over, so just prettyprint from here
                                      HERE
                                 else INNERLEFT]
                (PRIN1 ")"))
         (pop TAIL]))
```

(PROGV.PRETTYPRINT

```
[LAMBDA (EXPR)                                           ; Edited 31-Mar-88 11:30 by bvm
```

;; Prettyprinter advice for PROGV. Default way's main problem is that if the vars and values are non-lists the "body" of the form doesn't get
;; uniformly indented. Thus, you get something like (PROGV vars values (expression) <cr> (expression) ...)

```
    (if [OR (NLISTP (CDR EXPR))
            (LISTP (CADR EXPR))
            (NLISTP (CDR (LISTP (CDDR EXPR]
        then                                          ; 3 or fewer elements, or the second is a list--default prettyprinter
                                                      ; will do fine
             EXPR
      else (PRIN1 "(")
           (LET [(HERE (INDENTATION.FROM.HERE))
                 (LEFT (PROGN (PRIN2 (pop EXPR))      ; Print the car of form
                              (SPACES 1)
                              (DSPXPOSITION]
                 (DECLARE (SPECVARS LEFT))
                 (SUPERPRINT (CAR EXPR)
                        EXPR NIL *STANDARD-OUTPUT*)   ; Print the first element (vars) at this position
                 (pop EXPR)
                 (if (XCL::PPRINT-DEFINER-FITP (CAR EXPR))
                     then (SPACES 1)                  ; Room for next element (values) here
                          (SUPERPRINT (CAR EXPR)
                                 EXPR NIL *STANDARD-OUTPUT*)
                          (pop EXPR))                 ; Finally, print the body
                 (SEQUENTIAL.PRETTYPRINT EXPR HERE))
           (PRIN1 ")")                                ; Return NIL to say we handled it
           NIL])
```

(INDENTATION.FROM.HERE

```
  [LAMBDA NIL                                         ; Edited 28-Mar-88 18:17 by bvm
    ;; Returns X-pos about 3 chars over, for use in indenting code
    (+ (DSPXPOSITION)
       (TIMES 3 (CHARWIDTH (CHARCODE X)
                           *STANDARD-OUTPUT*])
```

(SEQUENTIAL.PRETTYPRINT

```
  [LAMBDA (TAIL LEFT)                                 ; Edited  1-Apr-88 14:12 by bvm
    (DECLARE (SPECVARS TAIL LEFT))
    ;; Print each element of tail indented at position LEFT.
    (PROG NIL
          (if (<= (DSPXPOSITION)
                  LEFT)
              then                                    ; Don't start with newline if we aren't to the right of the left margin
                   (GO MIDDLE))
      TOP (if (OR (NULL TAIL)
                  (PROGN (SUBPRINT/ENDLINE LEFT *STANDARD-OUTPUT*)
                         (NULL TAIL)))
              then                                    ; Done
                   (RETURN))
      MIDDLE
          (if (NLISTP TAIL)
              then                                    ; Degenerate tail
                   (RETURN (PRINTDEF TAIL T T T)))
          (SUPERPRINT (CAR TAIL)
                 TAIL NIL *STANDARD-OUTPUT*)
          (pop TAIL)
          (GO TOP])

)

(ADDTOVAR PRETTYPRINTMACROS
          (UNINTERRUPTABLY . CODEWRAPPER.PRETTYPRINT)
          (CL:UNWIND-PROTECT . CODEWRAPPER.PRETTYPRINT)
          (RESETLST . CODEWRAPPER.PRETTYPRINT)
          (CL:BLOCK . PROG1.PRETTYPRINT)
          (CL:IF . PROG1.PRETTYPRINT)
          (PROG1 . PROG1.PRETTYPRINT)
          (CL:WHEN . PROG1.PRETTYPRINT)
          (CL:UNLESS . PROG1.PRETTYPRINT)
          (WITH-READER-ENVIRONMENT . PROG1.PRETTYPRINT)
          (CL:CATCH . PROG1.PRETTYPRINT)
          (CASE . CASE.PRETTYPRINT)
          (CL:ECASE . CASE.PRETTYPRINT)
          (CL:ETYPECASE . CASE.PRETTYPRINT)
          (CL:TYPECASE . CASE.PRETTYPRINT)
          (CL:PROGV . PROGV.PRETTYPRINT)
          (WITH.MONITOR . PROG1.PRETTYPRINT))

(ADDTOVAR PRETTYEQUIVLST (PROG* . PROG)
                         (CL:COMPILER-LET . LET))

;; Repairs to other prettyprinting functions

(DEFINEQ
```

(SUPERPRINT/COMMENT

```
[LAMBDA (L FILE)                                          ; Edited 13-Apr-88 12:55 by bvm
  (DECLARE (USEDFREE LEFT TAIL RMARGIN FILEFLG MAKEMAP))
  (COND
     ((AND **COMMENT**FLG (NOT FILEFLG)
           (NOT MAKEMAP))
      ;; If we're eliding comments, not printing to a file, and not in DEdit, then just print the elision string
      (COND
         ((> (+ (DSPXPOSITION NIL FILE)
                (STRINGWIDTH **COMMENT**FLG FILE))
             (DSPRIGHTMARGIN NIL FILE))                   ; Watch out for overflowing the current line.
          (PRINENDLINE (DSPLEFTMARGIN NIL FILE)
                 FILE)))
      (PRIN1S **COMMENT**FLG NIL FILE))
     (T (PROG ((DSLMARG (DSPLEFTMARGIN NIL FILE))
               (HERE (DSPXPOSITION NIL FILE))
               (COMMENT-RMARGIN RMARGIN)
               (SEMIP (SEMI-COLON-COMMENT-P L))
               COMMENT-LMARGIN RIGHTFLG BODY HALFLINE)
              [if SEMIP
                  then                                    ; Extract the comment body
                       (COND
                          ((OR [NOT (STRINGP (SETQ BODY (CAR (LISTP (CDR (LISTP (CDR L]
                               (CDDDR L))                 ; Not a good semi-colon comment
                           (SETQ SEMIP NIL]
              [COND
                 [[SETQ RIGHTFLG (if SEMIP
                                     then                 ; Only 1-semi comments go in right margin
                                          (EQ SEMIP 1)
                                  else                    ; Short single * comments go at right
                                       (AND (NOT (SUPERPRINTEQ (CADR L)
                                                               COMMENTFLG))
                                            (<= (LENGTH L)
                                                15]       ; Print comment in the righthand margin
                  (SETQ COMMENT-LMARGIN (OR COMMENTCOL (SUPERPRINT/COMMENT1 L RMARGIN FILE]
                 ((AND SEMIP (NOT MAKEMAP))              ; Semi-colon comment > 1, unless under DEdit (lest we confuse
                                                         ; it)
                  (AND SEMIP (> SEMIP 2)
                       (NOT MAKEMAP))
                  (SETQ COMMENT-LMARGIN (if (EQ SEMIP 2)
                                            then         ; indent like code, but no more than a third of the way over if it
                                                         ; would take more than 2 lines to print this.
                                                 [MIN LEFT (MAX (- RMARGIN (FIXR (TIMES (STRINGWIDTH BODY FILE
                                                                                        )
                                                                                0.52)))
                                                                (+ DSLMARG (IQUOTIENT (- RMARGIN DSLMARG)
                                                                                      3]
                                          else           ; Comment should be printed flush left.
                                               DSLMARG)))
                 (T (LET ((INDENT (IQUOTIENT (- RMARGIN DSLMARG)
                                             11)))       ; Print old-style comment centered and wide, indented about
                                                         ; 10% from margins
                      (SETQ COMMENT-LMARGIN (+ DSLMARG INDENT))
                      (SETQ COMMENT-RMARGIN (- RMARGIN INDENT))
                      (COND
                         ((EQ HERE COMMENT-LMARGIN)
                          ;; HACK: Almost certainly called from REPP, so we must supress the normal leading and trailing blank lines as
                          ;; they have already been done
                          (SETQ RIGHTFLG T]
              (COND
                 ((AND (NULL RIGHTFLG)
                       (OR (NOT SEMIP)
                           (> SEMIP 1)))                 ; Centered comment starts on new line
                  (if (> HERE COMMENT-LMARGIN)
                      then                               ; We have not yet moved down a line, so do that first
                           (TERPRI FILE))
                  [if (AND (EQ SEMIP 2)
                           (IMAGESTREAMP FILE))
                      then                               ; For 2-semi comments, only go down half line, accomplished by
                                                         ; moving up half line now before this next endline
                           (RELMOVETO 0 (SETQ HALFLINE (IQUOTIENT (- (DSPLINEFEED NIL FILE))
                                                                  2]
                  (PRINENDLINE COMMENT-LMARGIN FILE))
                 ((< COMMENT-LMARGIN (DSPXPOSITION NIL FILE))  ; Past the starting point, so start new line
                  (PRINENDLINE COMMENT-LMARGIN FILE))
                 (T (DSPXPOSITION COMMENT-LMARGIN FILE)))
              (SETFONT (PROG1 (SETFONT COMMENTFONT FILE)
                              (COND
                                 ((AND SEMIP (NOT MAKEMAP)
                                       (OR *PRINT-SEMICOLON-COMMENTS* (IMAGESTREAMP FILE)))
                                                         ; do nice semi-colon stuff
                                  (PRIN2-LONG-STRING BODY FILE NIL NIL COMMENT-LMARGIN COMMENT-RMARGIN T SEMIP))
                                 (T                      ; Old comment or in DEdit (makemap true), so have to do it the
                                                         ; old way
                                    (SETQ SEMIP NIL)
                                    (SUPERPRINT/COMMENT2 L COMMENT-LMARGIN (IQUOTIENT (+ COMMENT-LMARGIN
```

```
                                                                                    COMMENT-RMARGIN)
                                                                                2)
                                        COMMENT-RMARGIN FILE))))
                    FILE)
             (if (OR (NULL SEMIP)
                     (> SEMIP 2))
                 then                                          ; Old centered comments and big semi-colon comments get new
                                                               ; line
                      (OR RIGHTFLG (PRINENDLINE DSLMARG FILE))
               elseif (NULL (CDR TAIL))
                 then                                          ; Nothing more will be printed.  So even though we were a short
                                                               ; comment, we need to go to new line so that the closing paren is
                                                               ; on a new line, rather than here after the comment (AR 8475)
                      (PRINENDLINE LEFT FILE)
               elseif [AND HALFLINE (NOT (AND (LISTP (CDR TAIL))
                                              (SEMI-COLON-COMMENT-P (LISTP (CADR TAIL]
                 then                                          ; Set off double-semi-colon comment by half line.  Don't do for
                                                               ; consecutive comments, since the next comment will take care
                                                               ; of it
                      (RELMOVETO 0 HALFLINE)
                      (PRINENDLINE DSLMARG FILE))
             (RETURN L])
```

(PRIN2-LONG-STRING

```
  [LAMBDA (STRING STREAM P2FLG TAIL LMARG RMARG COMMENTP USE-SEMI-COLONS)
                                                               ; Edited  4-Apr-88 14:26 by bvm
```

;; Fancy string printer that divides long strings into multiple lines at convenient breaks. If P2FLG is true, this is a call from PRIN2 or friend, in which ;; case the surrounding doublequotes are printed, as well as escapes in front of special chars. TAIL is the list car of which is STRING. LMARG ;; and RMARG specify the desired margins of the text. If COMMENTP is true, this is a comment. In addition, if USE-SEMI-COLONS is non-NIL, ;; this is a semi-colon comment with that many semis.

```
  (PROG ((ESC (fetch (READTABLEP ESCAPECHAR) of *READTABLE*))
         (SA (fetch (READTABLEP READSA) of *READTABLE*))
         (HERE (DSPXPOSITION NIL STREAM))
         (FONT (DSPFONT NIL STREAM))
         (IMSTREAMP (IMAGESTREAMP STREAM))
         ESCWIDTH SPACEWIDTH CLOSEWIDTH SEMIWIDTH LASTSPACE I C NEXTC POS J MAPX1 MAPY1 SINGLELEFT SEMISTRING
         ESCAPESEPRS SEMICLOSE)
        (COND
           ((NOT (type? FONTDESCRIPTOR FONT))                  ; Ugh, happens for files
            (SETQ FONT STREAM)))
        (SETQ ESCWIDTH (CHARWIDTH ESC FONT))
        (SETQ SPACEWIDTH (CHARWIDTH (CHARCODE SPACE)
                                    FONT))
        (SETQ CLOSEWIDTH (COND
                            (P2FLG (STRINGWIDTH "%")" FONT))
                            (T 0)))
        (if USE-SEMI-COLONS
            then (if (< USE-SEMI-COLONS 5)
                     then                                      ; Semicolon comment
                          [SETQ SEMIWIDTH (+ SPACEWIDTH (TIMES USE-SEMI-COLONS (CHARWIDTH (CHARCODE ";")
                                                                                          FONT]
                          (SETQ SEMISTRING (CONCAT (ALLOCSTRING USE-SEMI-COLONS (CHARCODE ";"))
                                                   " "))
                     else                                      ; Balanced (hash bar) comment
                          (SETQ SEMISTRING "#|")
                          (SETQ SEMIWIDTH (STRINGWIDTH SEMISTRING FONT))
                          (SETQ SEMICLOSE "|#")))
        [COND
           ((for C instring (PROGN                             ; dwimify bug tries to turn naked STRING into (STRING C) here.
                                   STRING)
               as I from 1 bind (POS _ (+ HERE (COND
                                                  (P2FLG (CHARWIDTH (CHARCODE %")
                                                                    FONT))
                                                  ((NULL USE-SEMI-COLONS)
                                                   0)
                                                  ((< USE-SEMI-COLONS 5)
                                                   SEMIWIDTH)
                                                  (T           ; Include the width of the closing |#
                                                     (TIMES 2 SEMIWIDTH)))
                                               CLOSEWIDTH))
               do (COND
                     ((EQ C (CHARCODE CR))                     ; Always want to print these strings specially
                      (SETQ LASTSPACE I)
                      (RETURN NIL))
                     ((AND P2FLG (OR (EQ C (CHARCODE %"))
                                     (EQ C ESC)))              ; Need escape
                      (add POS ESCWIDTH)))
                  (COND
                     ((> (add POS (CHARWIDTH C FONT))
                         RMARG)
                      (RETURN NIL)))
                  (COND
                     ((EQ C (CHARCODE SPACE))
                      (SETQ LASTSPACE I)))
               finally (RETURN T))                             ; It all fits on this line
```

```
            (RETURN (COND
                        (P2FLG (PRIN2S STRING TAIL STREAM))
                        (T (if SEMISTRING
                               then (PRIN1 SEMISTRING STREAM))
                           (PRIN1S STRING TAIL STREAM)
                           (if SEMICLOSE
                               then (PRIN1 SEMICLOSE STREAM]
         (COND
            ((OR (NULL LASTSPACE)
                 (AND (NULL COMMENTP)
                      (NEQ HERE LMARG)))
```

;; Can't print anything on this line before the end. Comments are allowed to have different first and subsequent margin.

```
             (PRINENDLINE (SETQ HERE LMARG)
                    STREAM)
             (SETQ LASTSPACE 0)))
         [COND
            (MAKEMAP                                          ; Note start
                   (SETQ MAPX1 HERE)
                   (SETQ MAPY1 (DSPYPOSITION NIL STREAM))
                   (SETQ SINGLELEFT (EQ HERE LMARG]
         [COND
            (P2FLG [COND
                       ((NOT (IMAGESTREAMP STREAM))           ; Need to be able to read it back
                        (SETQ ESCAPESEPRS T)
                        (LET ((HASH (fetch (READTABLEP HASHMACROCHAR) of *READTABLE*)))
                             (\OUTCHAR STREAM HASH)
                             (add HERE (CHARWIDTH HASH FONT]
                   (\OUTCHAR STREAM (CHARCODE %"))
                   (add HERE (CHARWIDTH (CHARCODE %")
                                        FONT)))
            (USE-SEMI-COLONS                                  ; Print the first set of semi-colons or #|
                   (PRIN1 SEMISTRING STREAM)
                   (add HERE SEMIWIDTH)
                   (if (EQ USE-SEMI-COLONS 5)
                       then                                   ; No more semis now
                            (SETQ SEMISTRING NIL]
```

;;; Now loop, printing as much as we can while there's room

```
         (SETQ I 0)
      LP [COND
            ([NULL (SETQ C (NTHCHARCODE STRING (add I 1]      ; Done
             (GO DONE))
            ((NOT (< I LASTSPACE))
```

;; Must find the next safe place to print up to. LASTSPACE is either a space or CR position, or is 0, which is our state when printing
;; from the left margin until we encounter a space.

```
             (SETQ POS HERE)
             (SETQ J I)                                       ; Ordinarily, J is pointing at a space or CR except when we have
                                                              ; just printed an endline
             (SELCHARQ C
                  (SPACE                                      ; Would like all spaces before the eol, where they're invisible, not
                                                              ; after
                         (SELCHARQ (NTHCHARCODE STRING (ADD1 J))
                              ((SPACE CR NIL)
                                   (SETQ LASTSPACE (ADD1 J))  ; Go ahead and print this space, and note that it is now okay to
                                                              ; break the line
                                   (COND
                                      ((AND (>= (+ HERE SPACEWIDTH)
                                                RMARG)
                                            (NOT ESCAPESEPRS)) ; Extra spaces have no effect, so don't print them at all, lest the
                                                              ; dsprightmargin bite
                                       (GO LP))
                                      (T (GO PRINTIT))))
                              NIL)
                         (add POS SPACEWIDTH))
                  (CR                                         ; If two cr's in a row, print them all; if only one, must escape it
                      (COND
                         ((EQ (SETQ C (NTHCHARCODE STRING (add I 1)))
                              (CHARCODE CR))
                          (PRINENDLINE LMARG STREAM)
                          (while (EQ (SETQ C (NTHCHARCODE STRING (add I 1)))
                                     (CHARCODE CR))
                             do (PRINENDLINE LMARG STREAM)))
                         (ESCAPESEPRS (\OUTCHAR STREAM ESC)))
                      (SETQ LASTSPACE 0)
                      (GO ENDLINE))
                  (PROGN ;; Gets set this way at left edge.  Must print something on this line, even if there are no spaces before the right edge
                         (GO CHECKESCAPE)))
             (SETQ LASTSPACE 0)
             (while (< POS RMARG) do (SELCHARQ (SETQ NEXTC (NTHCHARCODE STRING (add J 1)))
                                       ((CR SPACE)            ; Can safely go this far
                                            (SETQ LASTSPACE J)
                                            (RETURN))
                                       (NIL                   ; End of string -- ok if there is space for closing quote and paren
                                                              ; as well
```

```
                                                (COND
                                                   ((< (+ POS CLOSEWIDTH)
                                                       RMARG)
                                                    (SETQ LASTSPACE J)
                                                    (RETURN))
                                                   (T (GO $$OUT))))
                                             NIL)
                                          (COND
                                             ((OR (EQ NEXTC (CHARCODE %"))
                                                  (EQ NEXTC ESC))
                                              (add POS ESCWIDTH)))
                                          (add POS (CHARWIDTH NEXTC FONT))
                finally (COND
                           ((EQ LASTSPACE 0)                          ; Need break
                            (COND
                               [(EQ C (CHARCODE SPACE))               ; Will turn this space into CR
                                (SETQ C (NTHCHARCODE STRING (add I 1]
                               (T (SHOULDNT)))
                            (GO ENDLINE]
      CHECKESCAPE
          (COND
             ((AND P2FLG (OR (EQ C (CHARCODE %"))
                             (EQ C ESC)))
              (\OUTCHAR STREAM ESC)
              (add HERE ESCWIDTH)))
      PRINTIT
          (\OUTCHAR STREAM C)
          (add HERE (CHARWIDTH C FONT))
          (GO LP)
      ENDLINE
          (PRINENDLINE LMARG STREAM)
          (SETQ HERE LMARG)
          (COND
             ((NULL C)                                                ; Done
              (GO DONE))
             ((AND ESCAPESEPRS (EQ (\SYNCODE SA C)
                                   SEPRCHAR.RC))                      ; Have to quote sepr immediately following CR
              (\OUTCHAR STREAM ESC)
              (add HERE ESCWIDTH)
              (GO PRINTIT))
             (T (COND
                   (SEMISTRING (PRIN1 SEMISTRING STREAM)
                               (add HERE SEMIWIDTH)))
                (GO CHECKESCAPE)))
      DONE
          [COND
             (P2FLG (\OUTCHAR STREAM (CHARCODE %"]
          (COND
             [MAKEMAP (LET [(ENTRY (MAKEMAPENTRY TAIL (AND (NEQ MAKEMAP T)
                                                           MAKEMAP)
                                                 MAPX1 MAPY1 (DSPXPOSITION NIL STREAM)
                                                 (DSPYPOSITION NIL STREAM)
                                                 (\DEDITFONT# STREAM]
                           (replace LONGSTRINGP of ENTRY with T)
                           (COND
                              (SINGLELEFT (replace LONGSTRING1MARGINP of ENTRY with T)))
                           (COND
                              ((EQ (- (DSPRIGHTMARGIN NIL STREAM)
                                      LMARG)
                                   RMARG)
                               ;; Assume that RMARG not equal to stream's right margin only happens for centered comments.  In reality, it
                               ;; happens as well inside REPP, where RESETCLIP hides the true right margin.
                               (replace LONGSTRINGSYMMETRICP of ENTRY with T]
             (SEMICLOSE (PRIN1 SEMICLOSE STREAM)))
          (RETURN])
```

(SUPERPRINT/WRAPPER

```
  [LAMBDA (MACRO E TAIL BRFLG FILE)                                   ; Edited 31-Mar-88 12:00 by bvm
```

;;; Print E as MACRO followed by (CADR E), for example, print (QUOTE foo) as 'foo

```
    (PRINOPEN TAIL MACRO FILE)                                        ; Print the prefix
    [COND
       (MAKEMAP
```

;; Need to fool DEDIT into thinking that it is printing the whole list E when only (CADR E) appears in print. So do a fake entry for
;; (CAR E) whose width is zero

```
            (replace WRAPPER of MAKEMAP with MACRO)                   ; MAKEMAP is the entry for E -- want everyone to know it wasn't
                                                                      ; printed as normal list
            (LET ((X (DSPXPOSITION NIL FILE))
                  (Y (DSPYPOSITION NIL FILE)))
                 (MAKEMAPENTRY E (AND (NEQ MAKEMAP T)
                                      MAKEMAP)
                       X Y X Y (\DEDITFONT# FILE]
    (PROG1 (SUPERPRINT (CADR E)
```

```
                (CDR E)
                BRFLG FILE)                          ; Make sure to return the result of SUPERPRINT, so that caller
                                                     ; (eventually SUBPRINT) knows whether we printed something
                                                     ; like a list or not
        (PRINSHUT TAIL NIL FILE)                     ; Finally, print a vacuous closing paren

        )])
```

(SUPERPRINT/SPACE

```
  [LAMBDA (FILE)                                     ; Edited 31-Mar-88 12:18 by bvm
    ;; Print a space, preparing for next item to be printed
    (DECLARE (CL:SPECIAL RMARGIN SPACEWIDTH LEFT))   ; bound by prettyprinter stuff
    (if (< (- RMARGIN (DSPXPOSITION NIL FILE))
           (TIMES 2 SPACEWIDTH))
        then                                         ; printing a space will overflow the line, or if not then the next
                                                     ; char would, so go to new line
            (PRINENDLINE LEFT FILE)
      else (PRIN3 " " FILE])
```

(PRINENDLINE

```
  [LAMBDA (NEWXPOSITION FILE)                        ; Edited  1-Apr-88 14:24 by bvm
    ;; Terminate line, setting x at NEWXPOSITION.
    (OR FILE (SETQ FILE *STANDARD-OUTPUT*))
    (COND
       (MAKEMAP                                      ; From DEdit
              (MOVETO NEWXPOSITION (+ (DSPYPOSITION NIL FILE)
                                      (DSPLINEFEED NIL FILE))
                      FILE))
       (T (TERPRI FILE)
          (COND
             ((OR (SELECTQ (IMAGESTREAMTYPE FILE)
                       ((NIL TEXT)                   ; These don't know how to set x position
                           T)
                       (PROGN                        ; Assume all other image streams are ok
                              NIL))
                  (if (EQ FILE (TTYDISPLAYSTREAM))
                      then                           ; Even if FILE knows how to set xpos, the dribble file doesn't, so
                                                     ; use spaces
                          (DRIBBLEFILE)))
              (SETFONT (PROG1 (SETFONT DEFAULTFONT FILE)
                              ;; Print introductory spaces in the default font because we don't quite have this right yet for pspool files
                              (LET ((NS (QUOTIENT (- NEWXPOSITION (DSPXPOSITION NIL FILE))
                                                  SPACEWIDTH)))
                                   (RPTQ (QUOTIENT NS 8)
                                         (PRIN3 "        " FILE))
                                   (RPTQ (REMAINDER NS 8)
                                         (PRIN3 " " FILE))))
                      FILE)))
          (DSPXPOSITION NEWXPOSITION FILE])
```

(PRINDOTP

```
  [LAMBDA (E FILE)                                   ; Edited 13-Apr-88 15:08 by bvm
    ;; Print a dotted tail consisting of the non-list E, i.e., print " . <E>"
    (LET* [(DOT " . ")
           (MAXPOS (- RMARGIN (WIDTH E FILE T)
                      (WIDTH DOT FILE)
                      (WIDTH ")" FILE]                 ; MAXPOS is the rightmost position at which this will fit
           (if (AND (> (DSPXPOSITION NIL FILE)
                       MAXPOS)
                    (>= MAXPOS FIRSTPOS))
               then                                  ; Print dotted tail on next line as far to right as possible
                    (PRINENDLINE MAXPOS FILE))
           (PRIN3 DOT FILE)
           (PRIN2S E (COND
                        (MAKEMAP (MAKEDOTPTAIL E MAKEMAP))
                        (T (CONS E E)))
                 FILE])
```

(PRINTDEF1

```
  [LAMBDA (EXPR)                                     ; Edited  7-Apr-88 10:54 by bvm
    ;; Used by MAKEFILE to print P, etc expressions.  These are at top level, so must be forms!  But still print BLOCK: as a var to make it prettier
    (TERPRI)
    (PRINTDEF EXPR NIL (NEQ (CAR EXPR)
                            'BLOCK%:)
           NIL FNSLST)
    (TERPRI])
```

)

```
[XCL:REINSTALL-ADVICE 'MAKEFILE :AROUND '((:LAST (LET ((PRETTYFLG (AND (NOT (MEMB 'FAST OPTIONS))
                                                                       PRETTYFLG)))
                                                      (DECLARE (CL:SPECIAL PRETTYFLG))
                                                      *]

(READVISE MAKEFILE)

(DECLARE%: EVAL@COMPILE DOCOPY

(MOVD? '\DSPRETTY/ENDLINE 'SUBPRINT/ENDLINE NIL T)
)

(DECLARE%: EVAL@COMPILE DONTCOPY

(CL:PROCLAIM '(CL:SPECIAL **COMMENT**FLG *PRINT-SEMICOLON-COMMENTS* COMMENTFONT FNSLST RMARGIN SPACEWIDTH))

(FILESLOAD (LOADCOMP)
       DSPRINTDEF)
)

(DECLARE%: DONTEVAL@LOAD DOCOPY

(MOVD 'XCL::PPRINT-DEFINER 'PPRINT-DEFINER NIL T)
)

(PUTPROPS DEFINERPRINT FILETYPE :COMPILE-FILE)

(PUTPROPS DEFINERPRINT MAKEFILE-ENVIRONMENT (:PACKAGE "INTERLISP" :READTABLE "INTERLISP"))

(PUTPROPS DEFINERPRINT COPYRIGHT ("Xerox Corporation" 1988))
```

FUNCTION INDEX

PROPERTY INDEX

VARIABLE INDEX

ADVICE INDEX