File created: 21-Nov-2023 17:06:12 **{WMEDLEY}<library>POSTSCRIPTSTREAM.;12**

edit by: rmk

changes to: (FNS POSTSCRIPTFILEP)

previous date: 21-Jun-2021 20:29:32 {WMEDLEY}<library>POSTSCRIPTSTREAM.;11

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ POSTSCRIPTSTREAMCOMS
       [;; PostScript printer support for Medley
        (DECLARE%: EVAL@COMPILE DONTCOPY (RECORDS FONTID PSCFONT \POSTSCRIPTDATA POSTSCRIPTXFORM))
        (INITRECORDS \POSTSCRIPTDATA)
        (FNS POSTSCRIPT.INIT)
        (ADDVARS (DEFAULTFILETYPELIST (PS . TEXT)
                       (PSC . TEXT)
                       (PSF . BINARY)
                       (PSCFONT . BINARY)
                       (POSTSCRIPT . TEXT))
               (*DISPLAY-FONT-NAME-MAP* (AVANTGARDE-BOOK . AB)
                      (AVANTGARDE-DEMI . AD)
                      (BECKMAN . BM)
                      (BOOKMAN-LIGHT . BL)
                      (BOOKMAN-DEMI . BD)
                      (COURIER . CO)
                      (HELVETICA-NARROW . HN)
                      (NEWCENTURYSCHLBK . NC)
                      (PALATINO . PA)
                      (TIMES . TS)
                      (ZAPFCHANCERY-MEDIUM . ZM)
                      (ZAPFCHANCERY . ZC)
                      (ZAPFDINGBATS . ZD)))
        ;; Font-reading code
        (FNS PSCFONT.READFONT PSCFONT.SPELLFILE PSCFONT.COERCEFILE PSCFONTFROMCACHE.SPELLFILE
             PSCFONTFROMCACHE.COERCEFILE PSCFONT.WRITEFONT READ-AFM-FILE CONVERT-AFM-FILES POSTSCRIPT.GETFONTID
             POSTSCRIPT.FONTCREATE \POSTSCRIPT.SPECIALFONT.SCALEDWIDTHS POSTSCRIPT.FONTSAVAILABLE)
        (COMS ;; Until macro in FONT is exported
              (MACROS \FSETCHARWIDTH))
        (FNS OPENPOSTSCRIPTSTREAM CLOSEPOSTSCRIPTSTREAM)
        (INITVARS (*POSTSCRIPT-FILE-TYPE* 'BINARY))
        (FNS POSTSCRIPT.HARDCOPYW POSTSCRIPT.TEDIT POSTSCRIPT.TEXT POSTSCRIPTFILEP MAKEEPSFILE)
        (FNS POSTSCRIPT.BITMAPSCALE POSTSCRIPT.CLOSESTRING POSTSCRIPT.ENDPAGE POSTSCRIPT.OUTSTR
             POSTSCRIPT.PUTBITMAPBYTES POSTSCRIPT.PUTCOMMAND POSTSCRIPT.SET-FAKE-LANDSCAPE POSTSCRIPT.SHOWACCUM
             POSTSCRIPT.STARTPAGE \POSTSCRIPTTAB \PS.BOUTFIXP \PS.SCALEHACK \PS.SCALEREGION \SCALEDBITBLT.PSC
             \SETPOS.PSC \SETXFORM.PSC \STRINGWIDTH.PSC \SWITCHFONTS.PSC \TERPRI.PSC)
        ;; DIG operations:
        (FNS \BITBLT.PSC \BLTSHADE.PSC \CHARWIDTH.PSC \CREATECHARSET.PSC \DRAWARC.PSC \DRAWCIRCLE.PSC
             \DRAWCURVE.PSC \DRAWELLIPSE.PSC \DRAWLINE.PSC \DRAWPOINT.PSC \DRAWPOLYGON.PSC \DSPBOTTOMMARGIN.PSC
             \DSPCLIPPINGREGION.PSC \DSPCOLOR.PSC \DSPFONT.PSC \DSPLEFTMARGIN.PSC \DSPLINEFEED.PSC
             \DSPPUSHSTATE.PSC \DSPPOPSTATE.PSC \DSPRESET.PSC \DSPRIGHTMARGIN.PSC \DSPROTATE.PSC \DSPSCALE.PSC
             \DSPSCALE2.PSC \DSPSPACEFACTOR.PSC \DSPTOPMARGIN.PSC \DSPTRANSLATE.PSC \DSPXPOSITION.PSC
             \DSPYPOSITION.PSC \FILLCIRCLE.PSC \FILLPOLYGON.PSC \FIXLINELENGTH.PSC \MOVETO.PSC \NEWPAGE.PSC)
        (COMS ;; Character-output, plus special-cases:
              (FNS \POSTSCRIPT.CHANGECHARSET \POSTSCRIPT.OUTCHARFN \POSTSCRIPT.PRINTSLUG
                   \POSTSCRIPT.SPECIALOUTCHARFN \UPDATE.PSC \POSTSCRIPT.ACCENTFN \POSTSCRIPT.ACCENTPAIR)
              ;; Spacing-character (M-quad, etc.) and ballot-box-check &c special-case functions
              (FNS \PSC.SPACEDISP \PSC.SPACEWID \PSC.SYMBOLS)
              ;; The mapping of NS characters to Postscript renderings, both as an AList and as a hashtable
              (FNS \POSTSCRIPT.NSHASH)
              (VARS (*POSTSCRIPT-UNACCENTED-FONTS* '(Dancer ZapfDingbats "Dancer" "ZapfDingbats"))
                    *POSTSCRIPT-NS-TRANSLATIONS*)
              (GLOBALVARS *POSTSCRIPT-NS-HASH*))
        (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS \POSTSCRIPT.FRACTION))
        (VARS (\POSTSCRIPT.ORIENTATION.MENU (create MENU ITEMS _ '(("Landscape" T "Print this
                                                                   file/document/image in Landscape
                                                                   Orientation")
                                                                  ("Portrait" 'NIL "Print this
                                                                   file/document/image in Portrait
                                                                   Orientation"))
                                                 TITLE _ "Orientation" CENTERFLG _ T MENUOFFSET _
                                                 (create POSITION XCOORD _ -1 YCOORD _ 0)
                                                 CHANGEOFFSETFLG _ 'Y))
              (\POSTSCRIPT.ORIENTATION.OPTIONS.MENU (create MENU ITEMS _ '(("Ask" 'ASK "Always ask whether to
                                                                            print in Landscape or Portrait
                                                                            Orientation")
                                                                           ("Landscape" T "Default printing to
```

```
                                                                                    Landscape Orientation")
                                                                          ("Portrait" 'NIL "Default printing to
                                                                                    Portrait Orientation"))
                                                                TITLE _ "Default Orientation" CENTERFLG _ T))
      PS.BITMAPARRAY \POSTSCRIPT.JOB.SETUP SlopeMenuItems WeightMenuItems)
[ADDVARS (BackgroundMenuCommands ("PS Orientation" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE (MENU
                                                                                \POSTSCRIPT.ORIENTATION.OPTIONS.MENU
                                                                                ))
                                                "Select the default Orientation for PostScript output"
                                                (SUBITEMS ("Ask" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE 'ASK)
                                                                  "Always ask whether to print in Landscape or
                                                                  Portrait Orientation")
                                                       ("Landscape" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE T)
                                                              "Default printing to Landscape Orientation")
                                                       ("Portrait" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE NIL)
                                                              "Default printing to Portrait Orientation"]
(VARS (BackgroundMenu NIL))
(CONSTANTS (GOLDEN.RATIO 1.618034)
       (\PS.SCALE0 100)
       (\PS.TEMPARRAYLEN 20))
(INITVARS (POSTSCRIPT.BITMAP.SCALE 1)
       (POSTSCRIPT.EOL 'CR)
       (POSTSCRIPT.IMAGESIZEFACTOR 1)
       (POSTSCRIPT.PREFER.LANDSCAPE NIL)
       (POSTSCRIPT.TEXTFILE.LANDSCAPE NIL)
       (POSTSCRIPT.DEFAULT.PAGEREGION '(4800 4800 52800 70800))
       (POSTSCRIPT.TEXTURE.SCALE 4)
       [POSTSCRIPTFONTDIRECTORIES (LIST (COND ((EQ (MACHINETYPE)
                                                   'MAIKO)
                                               "{dsk}/USR/LOCAL/LDE/FONTS/POSTSCRIPT/")
                                              (T "{DSK}<LISPFILES>POSTSCRIPT>"]
       (\POSTSCRIPT.MAX.WILD.FONTSIZE 72))
[COMS (FNS POSTSCRIPTSEND)
      (ADDVARS (PRINTERTYPES ((POSTSCRIPT)
                              (CANPRINT (POSTSCRIPT))
                              (STATUS TRUE)
                              (PROPERTIES NILL)
                              (SEND POSTSCRIPTSEND)
                              (BITMAPSCALE POSTSCRIPT.BITMAPSCALE)
                              (BITMAPFILE (POSTSCRIPT.HARDCOPYW FILE BITMAP SCALEFACTOR REGION ROTATION
                                                  TITLE]
[ADDVARS (POSTSCRIPT.FONT.ALIST (HELVETICA . HELVETICA)
                (HELVETICAD . HELVETICA)
                (TIMESROMAN . TIMES)
                (TIMESROMAND . TIMES)
                (COURIER . COURIER)
                (GACHA . COURIER)
                (CLASSIC . NEWCENTURYSCHLBK)
                (MODERN . HELVETICA)
                (CREAM . HELVETICA)
                (TERMINAL . COURIER)
                (LOGO . HELVETICA)
                (OPTIMA . PALATINO)
                (TITAN . COURIER))
       [PRINTFILETYPES (POSTSCRIPT (TEST POSTSCRIPTFILEP)
                              (EXTENSION (PS PSC PSF))
                              (CONVERSION (TEXT POSTSCRIPT.TEXT TEDIT POSTSCRIPT.TEDIT]
       (IMAGESTREAMTYPES (POSTSCRIPT (OPENSTREAM OPENPOSTSCRIPTSTREAM)
                                 (FONTCREATE POSTSCRIPT.FONTCREATE)
                                 (FONTSAVAILABLE POSTSCRIPT.FONTSAVAILABLE)
                                 (CREATECHARSET \CREATECHARSET.PSC]
(INITVARS (POSTSCRIPT.PAGETYPE 'LETTER))
;; NIL means initial clipping is same as paper size.  Don't know why the other regions were specified--rmk
[APPENDVARS (POSTSCRIPT.PAGEREGIONS (LETTER (0 0 8.5 11)
                                            NIL
                                            (-0.1 -0.1 8.7 11.2))
                    (LEGAL (0 0 8.5 14)
                           NIL
                           (-0.1 -0.1 8.7 14.2))
                    (NOTE (0 0 8.5 11)
                          NIL
                          (-0.1 -0.1 8.7 11.2]
(GLOBALVARS DEFAULTPRINTINGHOST POSTSCRIPT.BITMAP.SCALE POSTSCRIPT.EOL POSTSCRIPT.FONT.ALIST
       POSTSCRIPT.PREFER.LANDSCAPE POSTSCRIPT.TEXTFILE.LANDSCAPE POSTSCRIPT.TEXTURE.SCALE
       POSTSCRIPTFONTDIRECTORIES \POSTSCRIPT.JOB.SETUP \POSTSCRIPT.MAX.WILD.FONTSIZE
       \POSTSCRIPT.ORIENTATION.MENU \POSTSCRIPTIMAGEOPS POSTSCRIPT.PAGETYPE POSTSCRIPT.PAGEREGIONS)
(DECLARE%: DONTEVAL@LOAD DOCOPY (P (POSTSCRIPT.INIT)))
(PROP (FILETYPE MAKEFILE-ENVIRONMENT)
      POSTSCRIPTSTREAM)
(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                       (NLAML)
                                                                       (LAMA POSTSCRIPT.PUTCOMMAND])
```

;; PostScript printer support for Medley

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

```
(DECLARE%: EVAL@COMPILE

(RECORD FONTID (FONTIDNAME FONTXFACTOR FONTOBLIQUEFACTOR))

(RECORD PSCFONT (FID IL-FONTID FIRSTCHAR LASTCHAR ASCENT DESCENT WIDTHS))

(DATATYPE \POSTSCRIPTDATA ((POSTSCRIPTACCENTED FLAG)            ; T if we're to do NS-to-PS translations on characters in the
                                                                ; current font.
                           POSTSCRIPTFONT                       ; The fontdescriptor of the current font
                           POSTSCRIPTX                          ; The current X
                           POSTSCRIPTY                          ; ... and Y
                           POSTSCRIPTLEFTMARGIN                 ; The margins
                           POSTSCRIPTRIGHTMARGIN POSTSCRIPTBOTTOMMARGIN POSTSCRIPTTOPMARGIN
                           POSTSCRIPTLINESPACING                ; Line to line spacing
                           POSTSCRIPTCOLOR                      ; Color (or grey shade) in effect; 0.0=black, 1.0=white.
                           POSTSCRIPTSCALE                      ; Scale of the stream
                           POSTSCRIPTOPERATION                  ; Default operation (PAINT, REPLACE, ...)
                           POSTSCRIPTCLIPPINGREGION             ; The current region available to be written into
                           POSTSCRIPTPAGENUM                    ; Current page number
                           POSTSCRIPTHEADING                    ; The heading
                           POSTSCRIPTHEADINGFONT                ; Font for the heading
                           POSTSCRIPTSPACEFACTOR                ; Expansion factor for spaces (see DSPSPACEFACTOR)
                           POSTSCRIPTSPACEWIDTH                 ; The width of a space in the current font
                           POSTSCRIPTLANDSCAPE                  ; non-NIL for paper in 'landscape' mode
                           POSTSCRIPTCHARSTOSHOW                ; non-NIL if the string (PostScript-type string) of chars has
                                                                ; already been started
                           POSTSCRIPTFONTCHANGEDFLG             ; Font has changed
                           POSTSCRIPTMOVEFLG                    ; Need to move
                           POSTSCRIPTWIDTHS                     ; The widths vector of the current font
                           POSTSCRIPTTRANSX                     ; Translation in X
                           POSTSCRIPTTRANSY                     ; ... and Y
                           POSTSCRIPTPENDINGXFORM               ; A userspace to devicespace transform is pending
                           POSTSCRIPTPAGEREGION                 ; The whole page
                           POSTSCRIPTPAGEBLANK                  ; This page is blank flag
                           POSTSCRIPTSCALEHACK                  ; For \PS.SCALEHACK since DSPSCALE doesn't change the
                                                                ; scale of the stream
                           POSTSCRIPTTEMPARRAY                  ; For converting FIXP to string of digit chars
                           POSTSCRIPTXFORMSTACK                 ; The stack of transformations.  DSPPUSHSTATE pushes one
                                                                ; onto this, DSPPOPSTATE uses it to reset values.
                           POSTSCRIPTROTATION                   ; Rotation value currently in effect.
                           POSTSCRIPTPENDINGROTATION            ; Rotation to take effect at next SETXFORM.
                           POSTSCRIPTFONTSUSED                  ; List of FONTIDs of the fonts that've been used before. This is
                                                                ; used to control the re-encoding of fonts for accented-character
                                                                ; rendering.
                           (POSTSCRIPTNSCHARSET BYTE)           ; Current NSCHARSET--widths are in POSTSCRIPTWIDTHS
                           (POSTSCRIPTNATURALSPACEWIDTH WORD)   ; Width of the space in the current font, used to compute the
                                                                ; scaled space width.

                           )
       POSTSCRIPTX _ 0 POSTSCRIPTY _ 0 POSTSCRIPTTRANSX _ 0 POSTSCRIPTTRANSY _ 0 POSTSCRIPTSPACEFACTOR _ 1
       POSTSCRIPTPAGENUM _ 0 POSTSCRIPTSCALEHACK _ 1 POSTSCRIPTTEMPARRAY _ (ARRAY \PS.TEMPARRAYLEN 'BYTE 0 0)
       POSTSCRIPTROTATION _ 0 POSTSCRIPTCOLOR _ 0.0)

(RECORD POSTSCRIPTXFORM (;; Holds the transformation state as saved by DSPPUSHSTATE.  Used by DSPPOPSTATE to restore the
                         ;; tranformation state.
                         PSXCLIP                                ; Clipping region
                         PSXPAGE                                ; Page region
                         PSXX                                   ; X position?
                         PSXY                                   ; Y position?
                         PSXLEFT                                ; Left margin
                         PSXRIGHT                               ; Right margin
                         PSXTOP                                 ; Top margin
                         PSXBOTTOM                              ; Bottom Margin
                         PSXTRANX                               ; X-translation in effect
                         PSXTRANY                               ; Y-translation in effect
                         PSXLAND                                ; Landscape?
                         PSXXFORMPEND                           ; Are there transforms pending?

                         ))
)

(/DECLAREDATATYPE '\POSTSCRIPTDATA
       '(FLAG POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER
              POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER
              POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER BYTE WORD)
       ;; ---field descriptor list elided by lister---
       '68)
)

(/DECLAREDATATYPE '\POSTSCRIPTDATA
       '(FLAG POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER
              POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER
              POINTER POINTER POINTER POINTER POINTER POINTER POINTER POINTER BYTE WORD)
       ;; ---field descriptor list elided by lister---
```

```
      '68)

(DEFINEQ

(POSTSCRIPT.INIT
  [LAMBDA NIL                                                    ; Edited 14-May-2018 10:48 by rmk:
                                                                 ; Edited  4-Feb-93 21:08 by jds
    (DECLARE (GLOBALVARS \POSTSCRIPT.CHARTYPE))

    ;; Add POSTSCRIPT font descriptions to the active font profile.

    [MAPC [CL:REMOVE-DUPLICATES (NCONC (for FD in FONTDEFS join (for FP in (CDR (ASSOC 'FONTPROFILE (CDR FD)))
                                                                  collect (CAR FP)))
                                       '(FONT7 FONT6 FONT5 FONT4 FONT3 FONT2 FONT1 BOLDFONT LITTLEFONT BIGFONT
                                               PRETTYCOMFONT COMMENTFONT USERFONT SYSTEMFONT CLISPFONT
                                               LAMBDAFONT CHANGEFONT DEFAULTFONT]
          (FUNCTION (LAMBDA (CLASS)
                      (LET (COPYFD OLDPSCFD)
                           (if (BOUNDP CLASS)
                               then (SETQ CLASS (EVALV CLASS))
                                    (if (TYPEP CLASS 'FONTCLASS)
                                        then (SETQ COPYFD (OR (fetch (FONTCLASS INTERPRESSFD) of CLASS)
                                                              (fetch (FONTCLASS PRESSFD) of CLASS)
                                                              (fetch (FONTCLASS DISPLAYFD) of CLASS)))
                                             (if (SETQ OLDPSCFD (ASSOC 'POSTSCRIPT (fetch (FONTCLASS OTHERFDS)
                                                                                       of CLASS)))
                                                 then [if (NOT (CDR OLDPSCFD))
                                                          then (RPLACD OLDPSCFD (if (LISTP COPYFD)
                                                                                    then COPYFD
                                                                                  else (FONTUNPARSE COPYFD]
                                               else (push (fetch (FONTCLASS OTHERFDS) of CLASS)
                                                          (CONS 'POSTSCRIPT (if (LISTP COPYFD)
                                                                                then COPYFD
                                                                              else (FONTUNPARSE COPYFD]
    [FOR FD IN FONTDEFS DO (FOR FP IN (CDR (ASSOC 'FONTPROFILE (CDR FD)))
                              DO (COND
                                   ((ASSOC 'POSTSCRIPT (CL:NTHCDR 5 FP))
                                    ;; There's already a postscript spec, so leave it be.

                                    )
                                   (T (NCONC1 FP `(POSTSCRIPT ,(OR (CL:FIFTH FP)
                                                                   (CL:FOURTH FP)
                                                                   (CL:THIRD FP]

    ;; Eliminate any existing postscript fonts, to start with a clean slate if reinitializing.

    (FOR FD IN (FONTSAVAILABLE '* '* '* '* 'POSTSCRIPT) DO (APPLY (FUNCTION SETFONTDESCRIPTOR)
                                                                  FD))
    (SETQ POSTSCRIPTFONTCACHE NIL)
    (SETQ \POSTSCRIPT.CHARTYPE (CL:MAKE-ARRAY 256 :INITIAL-ELEMENT T))

    ;; \POSTSCRIPT.OUTCHARFN uses this array to quickly determine whether a character needs any special processing -- T means yes

    (for x from (CHARCODE SP) to 126 unless (FMEMB x (CHARCODE (%( %) \))) do (CL:SETF (CL:AREF
                                                                                          \POSTSCRIPT.CHARTYPE
                                                                                          x)
                                                                                        NIL))

    ;; RMK:  Maybe the following is equivalent to alot of the stuff above??

    (FONTPROFILE.ADDDEVICE 'POSTSCRIPT 'INTERPRESS)
    (SETQ \POSTSCRIPTIMAGEOPS (create IMAGEOPS
                                      IMAGETYPE _ 'POSTSCRIPT
                                      IMCLOSEFN _ (FUNCTION CLOSEPOSTSCRIPTSTREAM)
                                      IMXPOSITION _ (FUNCTION \DSPXPOSITION.PSC)
                                      IMYPOSITION _ (FUNCTION \DSPYPOSITION.PSC)
                                      IMMOVETO _ (FUNCTION \MOVETO.PSC)
                                      IMFONT _ (FUNCTION \DSPFONT.PSC)
                                      IMLEFTMARGIN _ (FUNCTION \DSPLEFTMARGIN.PSC)
                                      IMRIGHTMARGIN _ (FUNCTION \DSPRIGHTMARGIN.PSC)
                                      IMLINEFEED _ (FUNCTION \DSPLINEFEED.PSC)
                                      IMDRAWLINE _ (FUNCTION \DRAWLINE.PSC)
                                      IMDRAWCURVE _ (FUNCTION \DRAWCURVE.PSC)
                                      IMDRAWCIRCLE _ (FUNCTION \DRAWCIRCLE.PSC)
                                      IMDRAWELLIPSE _ (FUNCTION \DRAWELLIPSE.PSC)
                                      IMFILLCIRCLE _ (FUNCTION \FILLCIRCLE.PSC)
                                      IMBLTSHADE _ (FUNCTION \BLTSHADE.PSC)
                                      IMBITBLT _ (FUNCTION \BITBLT.PSC)
                                      IMSCALEDBITBLT _ (FUNCTION \SCALEDBITBLT.PSC)
                                      IMNEWPAGE _ (FUNCTION \NEWPAGE.PSC)
                                      IMSCALE _ (FUNCTION \DSPSCALE.PSC)
                                      IMSCALE2 _ (FUNCTION \DSPSCALE2.PSC)
                                      IMCOLOR _ (FUNCTION \DSPCOLOR.PSC)
                                      IMTERPRI _ (FUNCTION \TERPRI.PSC)
                                      IMTOPMARGIN _ (FUNCTION \DSPTOPMARGIN.PSC)
                                      IMBOTTOMMARGIN _ (FUNCTION \DSPBOTTOMMARGIN.PSC)
                                      IMSPACEFACTOR _ (FUNCTION \DSPSPACEFACTOR.PSC)
                                      IMFONTCREATE _ 'POSTSCRIPT
                                      IMCLIPPINGREGION _ (FUNCTION \DSPCLIPPINGREGION.PSC)
                                      IMRESET _ (FUNCTION \DSPRESET.PSC)
```

```
                                    IMDRAWPOLYGON _ (FUNCTION \DRAWPOLYGON.PSC)
                                    IMFILLPOLYGON _ (FUNCTION \FILLPOLYGON.PSC)
                                    IMSTRINGWIDTH _ (FUNCTION \STRINGWIDTH.PSC)
                                    IMCHARWIDTH _ (FUNCTION \CHARWIDTH.PSC)
                                    IMDRAWARC _ (FUNCTION \DRAWARC.PSC)
                                    IMROTATE _ (FUNCTION \DSPROTATE.PSC)
                                    IMTRANSLATE _ (FUNCTION \DSPTRANSLATE.PSC)
                                    IMDRAWPOINT _ (FUNCTION \DRAWPOINT.PSC)
                                    IMPUSHSTATE _ (FUNCTION \DSPPUSHSTATE.PSC)
                                    IMPOPSTATE _ (FUNCTION \DSPPOPSTATE.PSC)))
    (SETQ *POSTSCRIPT-NS-HASH* (HARRAY 255))
    (\POSTSCRIPT.NSHASH *POSTSCRIPT-NS-TRANSLATIONS*])

)

(ADDTOVAR DEFAULTFILETYPELIST (PS . TEXT)
                              (PSC . TEXT)
                              (PSF . BINARY)
                              (PSCFONT . BINARY)
                              (POSTSCRIPT . TEXT))

(ADDTOVAR *DISPLAY-FONT-NAME-MAP* (AVANTGARDE-BOOK . AB)
                                  (AVANTGARDE-DEMI . AD)
                                  (BECKMAN . BM)
                                  (BOOKMAN-LIGHT . BL)
                                  (BOOKMAN-DEMI . BD)
                                  (COURIER . CO)
                                  (HELVETICA-NARROW . HN)
                                  (NEWCENTURYSCHLBK . NC)
                                  (PALATINO . PA)
                                  (TIMES . TS)
                                  (ZAPFCHANCERY-MEDIUM . ZM)
                                  (ZAPFCHANCERY . ZC)
                                  (ZAPFDINGBATS . ZD))

;; Font-reading code

(DEFINEQ
```

(PSCFONT.READFONT

```
  [LAMBDA (FONTFILENAME)                                    ; Edited  5-Oct-93 17:19 by rmk:
                                                            ; Edited  1-Sep-89 10:55 by jds
```

;; Read one of Matt Heffron's .PSC files, to get postscript font metrics. First check to see if incore cache as information indexed under the file's ;; name.

```
    (LET (FID W [S (OPENSTREAM FONTFILENAME 'INPUT NIL '((SEQUENTIAL T]
          (PF (create PSCFONT)))
      [replace (PSCFONT FID) of PF with (SETQ FID (READ S (FIND-READTABLE "INTERLISP"]
      ;; Read until we hit a 255 byte, marking the end of the font-id section.
      (CL:DO NIL
             ((EQ (BIN S)
                  255))
          ;; Body of the loop is empty, the test does all of the work

          )
      (replace (PSCFONT IL-FONTID) of PF with (CAR FID))
      (replace (PSCFONT FIRSTCHAR) of PF with (\WIN S))
      (replace (PSCFONT LASTCHAR) of PF with (\WIN S))
      (replace (PSCFONT ASCENT) of PF with (\WIN S))
      (replace (PSCFONT DESCENT) of PF with (\WIN S))
      (replace (PSCFONT WIDTHS) of PF with (SETQ W (ARRAY 256 'SMALLPOSP 0 0)))
      (for C from 0 to 255 do (SETA W C (\WIN S)))
      (CLOSEF S)
      ;; PATCH JDS 9/1/89:  The afm font reader made fonts too tall.  This should fix things pro tem.
      (replace (PSCFONT ASCENT) of PF with (- 1000 (fetch (PSCFONT DESCENT) OF PF)))
      (PUSH POSTSCRIPTFONTCACHE (CONS (L-CASE (FILENAMEFIELD FONTFILENAME 'NAME))
                                      (CREATE PSCFONT USING PF)))
      PF])
```

(PSCFONT.SPELLFILE

```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE)                ; Edited  5-Oct-93 22:15 by rmk:
                                                            ; Edited  5-Oct-92 15:23 by jds
```

;; Find the font file for a postscript font. Does the display-name conversion as well, for DOS.

```
    (CL:WHEN POSTSCRIPTFONTDIRECTORIES
        (\FINDFONTFILE (OR (CDR (FASSOC FAMILY POSTSCRIPT.FONT.ALIST))
                           FAMILY)
               SIZE FACE 0 DEVICE 0 POSTSCRIPTFONTDIRECTORIES '(PSCFONT PF PSC)))])
```

(PSCFONT.COERCEFILE

```
  [LAMBDA (FAMILY SIZE WEIGHT SLOPE EXPANSION ROTATION DEVICE)   ; Edited  5-Oct-93 16:28 by rmk:
```

;; This coerces the WEIGHT and SLOPE incrementally back to REGULAR in order to find a matching file.

```
    (COND
       ((AND (NEQ EXPANSION 'REGULAR)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST WEIGHT SLOPE 'REGULAR)
                     ROTATION DEVICE)))
       ((AND (EQ SLOPE 'ITALIC)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST WEIGHT 'REGULAR EXPANSION)
                     ROTATION DEVICE)))
       ((AND (NEQ EXPANSION 'REGULAR)
              (EQ SLOPE 'ITALIC)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST WEIGHT 'REGULAR 'REGULAR)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST 'MEDIUM SLOPE EXPANSION)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (NEQ EXPANSION 'REGULAR)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST 'MEDIUM SLOPE 'REGULAR)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (EQ SLOPE 'ITALIC)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST 'MEDIUM 'REGULAR EXPANSION)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (NEQ EXPANSION 'REGULAR)
              (EQ SLOPE 'ITALIC)
              (PSCFONT.SPELLFILE FAMILY SIZE (LIST 'MEDIUM 'REGULAR 'REGULAR)
                     ROTATION DEVICE])
```

(PSCFONTFROMCACHE.SPELLFILE

```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE)                    ; Edited  5-Oct-93 17:54 by rmk:
                                                                ; Edited  5-Oct-92 15:23 by jds
```

;; Tries to find postscript font information in the cache, indexed by the name-field of the fontfile.

```
    (LET [(CACHE (CDR (ASSOC (L-CASE (FILENAMEFIELD (\FONTFILENAME (OR (CDR (FASSOC FAMILY POSTSCRIPT.FONT.ALIST
                                                                                    ))
                                                                           FAMILY)
                                                                  SIZE FACE 'PSCFONT 0)
                                                    'NAME))
                              POSTSCRIPTFONTCACHE]
         (IF CACHE
             THEN (CREATE PSCFONT USING CACHE])
```

(PSCFONTFROMCACHE.COERCEFILE

```
  [LAMBDA (FAMILY SIZE WEIGHT SLOPE EXPANSION ROTATION DEVICE)   ; Edited  5-Oct-93 17:00 by rmk:
```

;; This coerces the WEIGHT and SLOPE incrementally back to REGULAR in order to find a matching font in the cache.

```
    (COND
       ((AND (NEQ EXPANSION 'REGULAR)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST WEIGHT SLOPE 'REGULAR)
                     ROTATION DEVICE)))
       ((AND (EQ SLOPE 'ITALIC)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST WEIGHT 'REGULAR EXPANSION)
                     ROTATION DEVICE)))
       ((AND (NEQ EXPANSION 'REGULAR)
              (EQ SLOPE 'ITALIC)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST WEIGHT 'REGULAR 'REGULAR)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST 'MEDIUM SLOPE EXPANSION)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (NEQ EXPANSION 'REGULAR)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST 'MEDIUM SLOPE 'REGULAR)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (EQ SLOPE 'ITALIC)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST 'MEDIUM 'REGULAR EXPANSION)
                     ROTATION DEVICE)))
       ((AND (NEQ WEIGHT 'MEDIUM)
              (NEQ EXPANSION 'REGULAR)
              (EQ SLOPE 'ITALIC)
              (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE (LIST 'MEDIUM 'REGULAR 'REGULAR)
                     ROTATION DEVICE])
```

(PSCFONT.WRITEFONT

```
  [LAMBDA (FONTFILENAME PF)                                     ; Edited  5-Aug-93 16:28 by sybalskY:MV:ENVOS
```

;; Given a PSCFONT data structure, write it out as a properly-named xxx.PSCFONT file, for later reading.

```
    NIL
    (LET ([S (OPENSTREAM FONTFILENAME 'OUTPUT NIL '((TYPE BINARY)
                                                    (SEQUENTIAL T]
          (W (fetch (PSCFONT WIDTHS) of PF))
          (*READTABLE* (FIND-READTABLE "INTERLISP")))
```

```
        (PRIN3 (fetch (PSCFONT FID) of PF)
               S)
        (BOUT S 0)
        (BOUT S 255)
        (\WOUT S (fetch (PSCFONT FIRSTCHAR) of PF))
        (\WOUT S (fetch (PSCFONT LASTCHAR) of PF))
        (\WOUT S (fetch (PSCFONT ASCENT) of PF))
        (\WOUT S (fetch (PSCFONT DESCENT) of PF))
        (for C from 0 to 255 do (\WOUT S (ELT W C)))
        (CLOSEF S)
        FONTFILENAME])
```

(READ-AFM-FILE

```
[LAMBDA (FILE BOLDNESS ITALICNESS)                    ; Edited 5-Aug-93 16:37 by sybalskY:MV:ENVOS
```

;; Read an Adobe-version-3 AFM file, and extract the metrics from it for making a PSCFONT file.

```
(LET ((IFILE (OPENSTREAM FILE 'INPUT))
      (PSCFONT (create PSCFONT))
      (FCHAR 1000)
      (LCHAR 0)
      (W (ARRAY 256 'SMALLPOSP 0 0))
      TOKEN WEIGHT SLOPE HEIGHT CMCOUNT FBBOX)
     (with PSCFONT PSCFONT (repeatuntil (STRING-EQUAL "FontName" (RSTRING IFILE)) do (READCCODE IFILE))
           (repeatwhile (STRING-EQUAL "" (SETQ TOKEN (RSTRING IFILE))) do (READCCODE IFILE))
           [COND
              ((NOT (AND (BOUNDP 'WeightMenu)
                         (type? MENU WeightMenu)))
               (SETQ WeightMenu (create MENU
                                        ITEMS _ WeightMenuItems
                                        MENUFONT _ (FONTCREATE 'HELVETICA 12]
           [COND
              ((NOT (AND (BOUNDP 'SlopeMenu)
                         (type? MENU SlopeMenu)))
               (SETQ SlopeMenu (create MENU
                                       ITEMS _ SlopeMenuItems
                                       MENUFONT _ (FONTCREATE 'HELVETICA 12]
           (OR (SETQ WEIGHT BOLDNESS)
               (printout T T "Font WEIGHT for " PSCFONT ": " (SETQ WEIGHT (MENU WeightMenu))
                     T))
           (OR (SETQ SLOPE ITALICNESS)
               (printout T T "Font SLOPE for " PSCFONT ": " (SETQ SLOPE (MENU SlopeMenu))
                     T))
           (SETQ FID (LIST TOKEN WEIGHT SLOPE 'REGULAR))
           [SETQ IL-FONTID (COND
                              ((AND (EQ SLOPE 'REGULAR)
                                    (EQ WEIGHT 'MEDIUM))
                               TOKEN)
                              (T (POSTSCRIPT.GETFONTID FID WEIGHT SLOPE 'REGULAR]
           [repeatuntil (STRING-EQUAL "StartCharMetrics" TOKEN)
              do (SETQ TOKEN (RSTRING IFILE))
                 (COND
                    [(STRING-EQUAL "FontBBox" TOKEN)
                     (SETQ FBBOX (LIST (READ IFILE)
                                       (READ IFILE)
                                       (READ IFILE)
                                       (READ IFILE)))
                     ;; The Ascender and Descender properties from the AFM file are currently ignored, and the values from the FontBBox
                     ;; are used, SCALED to the height of the font.
                     (SETQ DESCENT (IABS (CADR FBBOX)))
                     (SETQ ASCENT (CADDDR FBBOX))
                     (SETQ HEIGHT (IPLUS ASCENT DESCENT))
                     [SETQ DESCENT (FIXR (FTIMES DESCENT (/ 1000 HEIGHT]
                     (SETQ ASCENT (FIXR (FTIMES ASCENT (/ 1000 HEIGHT]
                    (T (READCCODE IFILE]
           (SETQ CMCOUNT (RATOM IFILE))
           (repeatuntil (EQ (CHARCODE EOL)
                            (READCCODE IFILE))
              do)
           (SETQ WIDTHS W)
           (for CC from 1 to CMCOUNT do (LET (CCODE)
                                             (repeatuntil (EQ 'C (RATOM IFILE)) do)
                                             (SETQ CCODE (READ IFILE))
                                             (RATOMS 'WX IFILE)
                                             (SETQ CWIDTH (READ IFILE))
                                             [COND
                                                ((CL:PLUSP CCODE)
                                                                ; This character appears in the standard encoding, so just use
                                                                ; the charcode.
                                                 (COND
                                                    ((ILESSP CCODE FCHAR)
                                                     (SETQ FCHAR CCODE)))
                                                 (COND
                                                    ((IGREATERP CCODE LCHAR)
                                                     (SETQ LCHAR CCODE)))
                                                 (SETA W CCODE CWIDTH))
```

```
                                                    (T              ; A character not in the standard encoding; look it up to see if it's
                                                                    ; one we need (eth & thorn are brought into the CS-0 codespace
                                                                    ; for UToronto's work).
                                                       (repeatuntil (EQ 'N (RATOM IFILE)) do
                                                                                    ;; Skip to the N entry, which gives the
                                                                                    ;; Adobe-standard name.
                                                                                          )
                                                       (SETQ CNAME (RATOM IFILE))
                                                                    ; GET THE NAME
                                                       (SETQ CCODE (LISTGET *POSTSCRIPT-EXTRA-CHARACTERS* CNAME))
                                                       (COND
                                                          (CCODE (COND
                                                                     ((ILESSP CCODE FCHAR)
                                                                      (SETQ FCHAR CCODE)))
                                                                 (COND
                                                                     ((IGREATERP CCODE LCHAR)
                                                                      (SETQ LCHAR CCODE)))
                                                                 (SETA W CCODE CWIDTH]
                                                 (repeatuntil (EQ (CHARCODE EOL)
                                                                  (READCCODE IFILE))
                                                    do)))
                (SETQ FIRSTCHAR FCHAR)
                (SETQ LASTCHAR LCHAR))
          (CLOSEF IFILE)
          PSCFONT])
```

(CONVERT-AFM-FILES

```
[LAMBDA (FILE-LIST)                                                 ; Edited  5-Aug-93 16:47 by sybalskY:MV:ENVOS
  (for FL in FILE-LIST do (LET ((FNAME (pop FL))
                                FONT FILENAME)
                               (for AFM-FILE in FL as WEIGHT in '(MEDIUM MEDIUM BOLD BOLD) as SLOPE
                                  in '(REGULAR ITALIC REGULAR ITALIC) do (SETQ FONT (READ-AFM-FILE AFM-FILE
                                                                                                  WEIGHT SLOPE))
                                                                         (SETQ FILENAME
                                                                          (\FONTFILENAME FNAME 1
                                                                                 (LIST WEIGHT SLOPE 'REGULAR)
                                                                                 'PSCFONT 0))
                                                                         (PSCFONT.WRITEFONT FILENAME FONT])
```

(POSTSCRIPT.GETFONTID

```
[LAMBDA (FID WEIGHT SLOPE EXPANSION)                                ; Edited 20-Nov-92 15:04 by sybalsky:mv:envos
  (LET (FONTID)
       (SETQ FONTID (create FONTID
                            FONTIDNAME _ (CAR FID)
                            FONTXFACTOR _ 1.0
                            FONTOBLIQUEFACTOR _ 0.0))
       [if (AND (NEQ (CADDR FID)
                     SLOPE)
                (EQ SLOPE 'ITALIC))
           then (replace (FONTID FONTOBLIQUEFACTOR) of FONTID with (CONSTANT (TAN 7.0]
       (if (AND (NEQ (CADR FID)
                     WEIGHT)
                (EQ WEIGHT 'BOLD))
           then                                                     ; Fake bold by slight expansion.
                (replace (FONTID FONTXFACTOR) of FONTID with 1.1))
       [if (NEQ EXPANSION 'REGULAR)
           then (replace (FONTID FONTXFACTOR) of FONTID with (TIMES (fetch (FONTID FONTXFACTOR) of FONTID)
                                                                    (if (EQ EXPANSION 'COMPRESSED)
                                                                        then (CONSTANT (QUOTIENT 1.0 GOLDEN.RATIO
                                                                                               ))
                                                                      else GOLDEN.RATIO]
       FONTID])
```

(POSTSCRIPT.FONTCREATE

```
[LAMBDA (FAMILY SIZE FACE ROTATION DEVICE)                          ; Edited 29-Oct-93 16:39 by rmk:
                                                                    ; Edited  3-Feb-93 17:22 by jds
  (LET (UNITFONT FULLNAME SCALEFONTP PSCFD ASCENT DESCENT FIXPWIDTHS PSCWIDTHSBLOCK WIDTHSBLOCK FD FACECHANGED
             (WEIGHT (CAR FACE))
             (SLOPE (CADR FACE))
             (EXPANSION (CADDR FACE)))
       ;; Ignore rotations, it is **MUCH** easier to rotate the Postscript stream user space coordinates.
       [COND
          [(EQ SIZE 1)
           ;; Since a 1 point font is ridiculously small, and it is the standard size for Postscript font info, a 1 point font is presumed to be the unit
           ;; size Postscript font info
           (COND
              ((SETQ PSCFD (PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE FACE ROTATION DEVICE))
               ;; Check in-core cache for exact match first
               (SETQ FACECHANGED NIL))
              ((SETQ FULLNAME (PSCFONT.SPELLFILE FAMILY SIZE FACE ROTATION DEVICE))
```

```
                    ;; Check file for exact match next
                    (SETQ PSCFD (PSCFONT.READFONT FULLNAME))
                    (SETQ FACECHANGED NIL))
                   ((SETQ PSCFD (PSCFONTFROMCACHE.COERCEFILE FAMILY SIZE WEIGHT SLOPE EXPANSION ROTATION DEVICE)
                    )
                    ;; Then check cache for coerced match
                    (SETQ FACECHANGED T))
                   ((SETQ FULLNAME (PSCFONT.COERCEFILE FAMILY SIZE WEIGHT SLOPE EXPANSION ROTATION DEVICE))
                    ;; Check file for coerced match
                    (SETQ PSCFD (PSCFONT.READFONT FULLNAME))
                    (SETQ FACECHANGED T)))
                (COND
                   (PSCFD (SETQ ASCENT (FIXR (TIMES (fetch (PSCFONT ASCENT) of PSCFD)
                                                    0.1)))
                          (SETQ DESCENT (FIXR (TIMES (fetch (PSCFONT DESCENT) of PSCFD)
                                                     0.1)))
                          (COND
                             (FACECHANGED (replace (PSCFONT IL-FONTID) of PSCFD with (POSTSCRIPT.GETFONTID
                                                                                       (fetch (PSCFONT FID)
                                                                                          of PSCFD)
                                                                                       WEIGHT SLOPE EXPANSION]
             ((SETQ UNITFONT (FONTCREATE FAMILY 1 FACE ROTATION DEVICE T))
              (SETQ PSCFD (LISTGET (fetch (FONTDESCRIPTOR OTHERDEVICEFONTPROPS) of UNITFONT)
                                   'PSCFONT))
              ;; Scale the ASCENT and DESCENT
              (SETQ ASCENT (FIXR (TIMES SIZE (fetch (PSCFONT ASCENT) of PSCFD)
                                        0.1)))
              (SETQ DESCENT (FIXR (TIMES SIZE (fetch (PSCFONT DESCENT) of PSCFD)
                                         0.1)))
              (SETQ SCALEFONTP T))
             (T ;; Here for fonts that only come in specific sizes.  Their info is not scaled like built-in Postscript fonts, it is already correct for this
                ;; pointsize.
                (COND
                   ([SETQ PSCFD (COND
                                   ((PSCFONTFROMCACHE.SPELLFILE FAMILY SIZE FACE ROTATION DEVICE))
                                   ((SETQ FULLNAME (PSCFONT.SPELLFILE FAMILY SIZE FACE ROTATION DEVICE))
                                    (PSCFONT.READFONT FULLNAME]
                    (SETQ ASCENT (fetch (PSCFONT ASCENT) of PSCFD))
                    (SETQ DESCENT (fetch (PSCFONT DESCENT) of PSCFD))
                    (SETQ SCALEFONTP NIL]
(COND
   (PSCFD ;; Set up the Charset descriptions and Widths vectors for character set 0:
          (SETQ FD
           (create FONTDESCRIPTOR
                   OTHERDEVICEFONTPROPS _ (LIST 'PSCFONT PSCFD)
                   FONTSCALE _ 100
                   FONTDEVICE _ DEVICE
                   FONTFAMILY _ FAMILY
                   FONTSIZE _ SIZE
                   FONTFACE _ FACE
                   ROTATION _ 0
                   \SFHeight _ (IPLUS ASCENT DESCENT)
                   \SFAscent _ ASCENT
                   \SFDescent _ DESCENT))
          (SETQ WIDTHSBLOCK (fetch (CHARSETINFO WIDTHS) of (\GETCHARSETINFO 0 FD)))
          (SETQ FIXPWIDTHS (fetch (PSCFONT WIDTHS) of PSCFD))
          [COND
             [SCALEFONTP (for CH from 0 to 255 do (\FSETWIDTH WIDTHSBLOCK CH
                                                              (FIXR (TIMES SIZE (ELT FIXPWIDTHS CH)
                                                                           0.1]
             (T (for CH from 0 to 255 do (\FSETWIDTH WIDTHSBLOCK CH (ELT FIXPWIDTHS CH]
          (SETQ PSCWIDTHSBLOCK (\CREATECSINFOELEMENT))
          ;; PSCWIDTHSBLOCK preserves the scaled widths from the original postscript metrics, not the NS mapping of them, which
          ;; goes into WIDTHSBLOCK.
          (for CH from 0 to 255 do (\FSETWIDTH PSCWIDTHSBLOCK CH (\FGETWIDTH WIDTHSBLOCK CH)))
          [LET [(TMP (COND
                        (FULLNAME (\FONTINFOFROMFILENAME FULLNAME DEVICE))
                        (UNITFONT (fetch FONTDEVICESPEC of UNITFONT]
               ;; If face got coerced (possibly in recursive call for unit font) then set FONTDEVICESPEC to describe what we really got
               (COND
                  ((AND TMP (NEQ FAMILY (CAR TMP)))
                   (replace FONTDEVICESPEC of FD with (LIST (CAR TMP)
                                                            SIZE
                                                            (COPY FACE)
                                                            0 DEVICE]
          [LET ((SYMWIDTHS (\POSTSCRIPT.SPECIALFONT.SCALEDWIDTHS 'SYMBOL FD ROTATION DEVICE))
                (DINGWIDTHS (\POSTSCRIPT.SPECIALFONT.SCALEDWIDTHS 'ZAPFDINGBATS FD ROTATION DEVICE)))
               ;; Now run thru the mapping table, filling in the new font from whatever source is specified:
               [MAPHASH *POSTSCRIPT-NS-HASH*
```

```
                                (FUNCTION (LAMBDA (MAPPING CODE)
                                                  (DESTRUCTURING-BIND
                                                   (KIND CODE2 BASECHAR)
                                                   MAPPING
                                                   ;; Depending on what kind of item it is, process it:
                                                   (SELECTQ KIND
                                                        (NIL ;; Translating an NS character to a PSC char in CS 0.
                                                             (\FSETCHARWIDTH FD CODE (\FGETWIDTH PSCWIDTHSBLOCK
                                                                                                (\CHAR8CODE CODE2))))
                                                        (SYMBOL [AND SYMWIDTHS (\FSETCHARWIDTH FD CODE
                                                                                               (ELT SYMWIDTHS (\CHAR8CODE CODE2]
)
                                                        (DINGBAT [AND DINGWIDTHS (\FSETCHARWIDTH FD CODE
                                                                                            (ELT DINGWIDTHS (\CHAR8CODE
                                                                                                             CODE2])
                                                        (FUNCTION ;; This is fake and only works for the fractions.  Need a better case.
                                                                  [\FSETCHARWIDTH
                                                                   FD CODE (IPLUS (\FGETWIDTH PSCWIDTHSBLOCK 164)
                                                                                  (FIXR (FTIMES 1.3 (\FGETWIDTH
                                                                                                     PSCWIDTHSBLOCK
                                                                                                     (CHARCODE 1])
                                                        (ACCENT          ; CODE2 is the rendering character but width comes from width
                                                                         ; of basechar
                                                                (\FSETCHARWIDTH FD CODE (\FGETWIDTH PSCWIDTHSBLOCK
                                                                                                    BASECHAR)))
                                                        (ACCENTPAIR
                                                                 ;; CODE2 and BASECHAR are overprinted, width is taken from CODE2
                                                                 ;; (the real character), basechar is the accent
                                                            (\FSETCHARWIDTH FD CODE (\FGETWIDTH PSCWIDTHSBLOCK
                                                                                               CODE2)))
                                                        (PROGN
                                                   ;; Skip APPLY*'s on this pass, waiting until normal characters get set up, so that widths of other NS
                                                   ;; characters are available.  Also skip anything else
                                                                NIL]
                              ;; Now do APPLY*'s.  MAPPING is of the form ('APPLY* DATA PRINTFN WIDTHFN). WIDTHFN gets applied to FD and
                              ;; DATA (coerced by INITFN)
                              (MAPHASH *POSTSCRIPT-NS-HASH* (FUNCTION (LAMBDA (MAPPING CODE)
                                                                              (CL:WHEN (EQ (CAR MAPPING)
                                                                                           'APPLY*)
                                                                                  (\FSETCHARWIDTH FD CODE
                                                                                         (APPLY* (CADDDR MAPPING)
                                                                                                 FD
                                                                                                 (CADR MAPPING))))]
                                    FD)
                       (T NIL])
```

(**POSTSCRIPT.SPECIALFONT.SCALEDWIDTHS**

```
  [LAMBDA (TYPE FD ROTATION DEVICE)                          ; Edited  5-Oct-93 18:21 by rmk:
    ;; Returns the scaled widths for a unit font of type TYPE (SYMBOL or ZAPFDINGBATS) compatible with FD.  A separate function so that the unit
    ;; widths can be easily cached.
    (LET [TYPEFONT WIDTHS NEWWIDTHS (SIZE (FETCH FONTSIZE OF FD))
               (FONTFILE (OR (PSCFONT.SPELLFILE TYPE 1 (FETCH (FONTDESCRIPTOR FONTFACE) OF FD)
                                                ROTATION DEVICE)
                             (PSCFONT.SPELLFILE 'SYMBOL 1 '(MEDIUM REGULAR REGULAR)
                                                ROTATION DEVICE]
         [SETQ TYPEFONT (COND
                           ((PSCFONTFROMCACHE.SPELLFILE TYPE 1 (FETCH (FONTDESCRIPTOR FONTFACE) OF FD)
                                  ROTATION DEVICE))
                           ((SETQ FONTFILE (PSCFONT.SPELLFILE TYPE 1 (FETCH (FONTDESCRIPTOR FONTFACE)
                                                                            OF FD)
                                                              ROTATION DEVICE))
                            (PSCFONT.READFONT FONTFILE))
                           ((PSCFONTFROMCACHE.SPELLFILE 'SYMBOL 1 '(MEDIUM REGULAR REGULAR)
                                  ROTATION DEVICE))
                           ((SETQ FONTFILE (PSCFONT.SPELLFILE 'SYMBOL 1 '(MEDIUM REGULAR REGULAR)
                                                              ROTATION DEVICE))
                            (PSCFONT.READFONT FONTFILE]
         (CL:WHEN (AND TYPEFONT (SETQ WIDTHS (FETCH (PSCFONT WIDTHS) OF TYPEFONT)))
             (SETQ NEWWIDTHS (ARRAY 256 'SMALLPOSP 0 0))
             ;; Have to copy because of scaling
             [FOR CH FROM 0 TO 255 DO (SETA NEWWIDTHS CH (FIXR (TIMES SIZE (ELT WIDTHS CH)
                                                                      0.1]
             NEWWIDTHS)])
```

(**POSTSCRIPT.FONTSAVAILABLE**

```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE)                 ; Edited 12-Jan-88 13:04 by Matt Heffron
    ;; the filtering code was borrowed from Richard Burton's \SEARCHINTERPRESSFONTS.  Note that without it [HELVETICA * (MEDIUM REGULAR
```

;; REGULAR)] would pick up [HELVETICA-NARROW * (MEDIUM REGULAR REGULAR)] as well.

```
(LET ((PATTERN (\FONTFILENAME (OR (CDR (ASSOC FAMILY POSTSCRIPT.FONT.ALIST))
                                  FAMILY)
                              SIZE FACE 'PSCFONT))
      [INVERSE.ALIST (for PAIR in POSTSCRIPT.FONT.ALIST collect (CONS (CDR PAIR)
                                                                      (CAR PAIR]
      FONTSAVAILABLE)
     (SETQ FONTSAVAILABLE
      (for FD in [for DIRECTORY in POSTSCRIPTFONTDIRECTORIES
                  join (for FILE in (DIRECTORY (CONCAT DIRECTORY PATTERN))
                          collect (LET* ((RAWFD (\FONTINFOFROMFILENAME FILE DEVICE))
                                         (RAWNAME (CAR RAWFD)))
                                    (RPLACA RAWFD (OR (CDR (ASSOC RAWNAME INVERSE.ALIST))
                                                      RAWNAME]
         when (AND (OR (EQ FAMILY '*)
                       (EQ FAMILY (CAR FD)))
                   (OR (EQ SIZE '*)
                       (EQ SIZE (CADR FD))
                       (EQ (CADR FD)
                           1))
                   (OR (EQ FACE '*)
                       (EQUAL FACE (CADDR FD))
                       (EQUAL [CDR (ASSOC FACE '((MRR MEDIUM REGULAR REGULAR)
                                                 (STANDARD MEDIUM REGULAR REGULAR)
                                                 (MIR MEDIUM ITALIC REGULAR)
                                                 (ITALIC MEDIUM ITALIC REGULAR)
                                                 (BRR BOLD REGULAR REGULAR)
                                                 (BOLD BOLD REGULAR REGULAR)
                                                 (BIR BOLD ITALIC REGULAR)
                                                 (BOLDITALIC BOLD ITALIC REGULAR]
                                   (CADDR FD)))
                   (NOT (MEMBER FD $$VAL)))
         collect FD))
      (if (EQ SIZE '*)
          then
```

;;; If SIZE was wildcarded, then provide list of pointsizes for Postscript scaled fonts (those with a 1 point descriptor file)

```
                (for FD in FONTSAVAILABLE
                   join (if (EQ 1 (CADR FD))
                            then (CONS FD (for NF in (for S from 2 to \POSTSCRIPT.MAX.WILD.FONTSIZE
                                                        collect (LET ((NFD (COPY FD)))
                                                                  (RPLACA (CDR NFD)
                                                                          S)
                                                                  NFD))
                                               unless (MEMBER NF FONTSAVAILABLE) collect NF))
                            else (LIST FD)))
          else FONTSAVAILABLE])

)

;; Until macro in FONT is exported

(DECLARE%: EVAL@COMPILE

(PUTPROPS \FSETCHARWIDTH MACRO (OPENLAMBDA (FONTDESC CHARCODE WIDTH)
                                 (\FSETWIDTH (ffetch (CHARSETINFO WIDTHS) of (\GETCHARSETINFO (\CHARSET CHARCODE
                                                                                                )
                                                                                    FONTDESC))
                                             (\CHAR8CODE CHARCODE)
                                             WIDTH)))
)

(DEFINEQ

(OPENPOSTSCRIPTSTREAM
  [LAMBDA (FILE OPTIONS)                                    ; Edited 12-Jun-2021 19:14 by rmk:
                                                            ; Edited 31-May-93 12:42 by sybalsky:mv:envos
                                                            ; Edited 23-Dec-92 01:17 by jds
```

;; RMK: Note: At open, this does a lot of printing using generic functions which invoke the generic \OUTCHARFN of the stream. We set that up as ;; BOUT. But after the stream is open, we install the \POSTSCRIPT.OUTCHARFN, below. We also have to make sure that other internal printing ;; that may want to use generic functions (PRIN1, PRIN3...) for convenience, doesn't cycle through the postscript outcharfn.

```
    (LET [[STREAM (OPENSTREAM (PACKFILENAME 'BODY FILE 'EXTENSION 'PS)
                              'OUTPUT NIL `((TYPE ,*POSTSCRIPT-FILE-TYPE*)
                                            (SEQUENTIAL T]
          (IMAGEDATA (create \POSTSCRIPTDATA))
          PAPER IMAGESIZEFACTOR CLIP REG (BBOX (LISTGET OPTIONS 'BOUNDINGBOX]
         (replace (STREAM IMAGEDATA) of STREAM with IMAGEDATA)
         (replace (STREAM IMAGEOPS) of STREAM with \POSTSCRIPTIMAGEOPS)
         (replace (STREAM OUTCHARFN) OF STREAM WITH (FUNCTION BOUT))
```

;; Bounding box is for encapsulated postscript. The bounding box is in Medley's postscript-coordinate system, so we have to scale it back to ;; default postscript since it will be interpreted outside of the operators specified below. CEIL and FLOOR to make sure that we don't leave ;; anything out. We may also want to change the header to have the EPSF qualifier

```
         (printout STREAM "%%!PS-Adobe-2.0" T %# (CL:WHEN BBOX
```

```
                                                    (PRINTOUT STREAM "%%%%BoundingBox: " (CL:FLOOR (CAR BBOX)
                                                                                                  \PS.SCALE0)
                                                           " "
                                                           (CL:FLOOR (CADR BBOX)
                                                                  \PS.SCALE0)
                                                           " "
                                                           (CL:CEILING (CADDR BBOX)
                                                                  \PS.SCALE0)
                                                           " "
                                                           (CL:CEILING (CADDDR BBOX)
                                                                  \PS.SCALE0)
                                                           T))
       "%%%%Title: "
       (MKSTRING (OR (LISTGET OPTIONS 'DOCUMENT.NAME)
                     FILE))
       T "%%%%Creator: PostScript Driver Copyright (C) 1988-1992 Venue and others" T
       "%%%%CreationDate: " (DATE)
       T %# (COND
               ((EQ 'LPT (FILENAMEFIELD STREAM 'HOST))
                ;; Put current user's name on break page only if going to LPT for immediate printing.  Presumably the print-spooler
                ;; itself should know what the user's system login-name is, but that may not be the case for all printers in all
                ;; environments.
                (PRINTOUT NIL "%%%%For: " (MKSTRING USERNAME)
                          T)))
       "%%%%EndComments" T)
(for X in \POSTSCRIPT.JOB.SETUP do (POSTSCRIPT.OUTSTR STREAM X)
                                   (\BOUTEOL STREAM))
(SETQ PAPER (OR (CDR (CL:ASSOC (SETQ PAPER (OR (LISTGET OPTIONS 'PAGETYPE)
                                               (LISTGET OPTIONS 'PAPERTYPE)
                                               POSTSCRIPT.PAGETYPE))
                               POSTSCRIPT.PAGEREGIONS :TEST #'STRING-EQUAL))
                (ERROR "Unknown PostScript page type" PAPER)))
```

;; Set the paper size:

```
(PRINTOUT STREAM (L-CASE (OR (LISTGET OPTIONS 'PAGETYPE)
                             (LISTGET OPTIONS 'PAPERTYPE)
                             POSTSCRIPT.PAGETYPE))
      T)
(COND
   ((NOT (AND [SETQ IMAGESIZEFACTOR (NUMBERP (LISTGET OPTIONS 'IMAGESIZEFACTOR]
               (CL:PLUSP IMAGESIZEFACTOR)))
    (SETQ IMAGESIZEFACTOR 1)))
[COND
   ((AND (NUMBERP POSTSCRIPT.IMAGESIZEFACTOR)
         (CL:PLUSP POSTSCRIPT.IMAGESIZEFACTOR))
    (SETQ IMAGESIZEFACTOR (TIMES IMAGESIZEFACTOR POSTSCRIPT.IMAGESIZEFACTOR]
(printout STREAM "/imagesizefactor " IMAGESIZEFACTOR " def" T)
(printout STREAM "%%%%EndSetup" T)
(replace (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA with \PS.SCALE0)
(replace (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA with (\PS.SCALEREGION (/ (TIMES 72
                                                                                              \PS.SCALE0)
                                                                                      IMAGESIZEFACTOR)
                                                                        (CAR PAPER)))
```

;; Initial clipping region can be specified separately from the page size, default is to page size.

```
[replace (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA with (SETQ CLIP
                                                                         (\PS.SCALEREGION
                                                                          (/ (TIMES 72 \PS.SCALE0)
                                                                             IMAGESIZEFACTOR)
                                                                          (OR (CADR PAPER)
                                                                              (CAR PAPER]
```

;; If a REGION parameter was supplied, it establishes the initial margins.

```
(SETQ REG (OR (AND (SETQ REG (LISTGET OPTIONS 'REGION))
                   (INTERSECTREGIONS REG CLIP))
              (CREATEREGION 3600 3600 54000 72000)))
(replace (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA with (fetch (REGION LEFT) of REG))
(replace (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA with (fetch (REGION BOTTOM) of REG))
(replace (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA with (PLUS (fetch (REGION BOTTOM) of REG)
                                                                        (fetch (REGION HEIGHT) of REG)
                                                                        -1))
(replace (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA with (PLUS (fetch (REGION LEFT) of REG)
                                                                         (fetch (REGION WIDTH)
                                                                            of REG)
                                                                         -1))
(\DSPFONT.PSC STREAM (FONTCREATE (OR [CAR (MKLIST (LISTGET OPTIONS 'FONTS]
                                                  DEFAULTFONT)
                                     NIL NIL NIL STREAM))
(\SWITCHFONTS.PSC STREAM IMAGEDATA)
[COND
   ((replace (\POSTSCRIPTDATA POSTSCRIPTHEADING) of IMAGEDATA with (LISTGET OPTIONS 'HEADING))
    (replace (\POSTSCRIPTDATA POSTSCRIPTHEADINGFONT) of IMAGEDATA with (COND
                                                                         ((LISTGET OPTIONS 'HEADINGFONT)
                                                                          (FONTCREATE
                                                                           (LISTGET OPTIONS
                                                                                  'HEADINGFONT)
```

```
                                                                NIL NIL NIL STREAM))
                                                              (T (fetch (\POSTSCRIPTDATA
                                                                         POSTSCRIPTFONT)
                                                                   of IMAGEDATA]
```

;; Decide if it's landscape: if (LANDSCAPE T) appears in OPTIONS, it is. IF ROTATION isn't DEFAULT, it is.

```
(COND
   ([COND
       ((CL:GETF OPTIONS 'LANDSCAPE NIL))
       ((EQL (CL:GETF OPTIONS 'ROTATION 'DEFAULT)
             'DEFAULT)
        (COND
           ((EQL POSTSCRIPT.PREFER.LANDSCAPE 'ASK)
            (MENU \POSTSCRIPT.ORIENTATION.MENU))
           (T POSTSCRIPT.PREFER.LANDSCAPE)))
       (T (CL:GETF OPTIONS 'ROTATION]
    (POSTSCRIPT.SET-FAKE-LANDSCAPE STREAM 90)))
```

;; Now we are ready for callers to use generic functions--see note above. The special external format ensures that e.g. COPYCHARS won't
;; do COPYBYTES when copying from a text file to a PS stream.

```
(\EXTERNALFORMAT STREAM (CREATE EXTERNALFORMAT
                                NAME _ 'POSTSCRIPT
                                OUTCHARFN _ (FUNCTION \POSTSCRIPT.OUTCHARFN)
                                EOL _ (FETCH (STREAM EOLCONVENTION) OF STREAM)))
(POSTSCRIPT.STARTPAGE STREAM)
STREAM])
```

(CLOSEPOSTSCRIPTSTREAM

```
  [LAMBDA (STREAM)                                          ; Edited  8-Mar-93 10:31 by jds
    (POSTSCRIPT.ENDPAGE STREAM)
    (POSTSCRIPT.PUTCOMMAND STREAM :EOL "%%%%Trailer" :EOL)     (* BOUT STREAM (CHARCODE ^D))
    ])

)

(RPAQ? *POSTSCRIPT-FILE-TYPE* 'BINARY)

(DEFINEQ
```

(POSTSCRIPT.HARDCOPYW

```
  [LAMBDA (FILE BITMAP SCALEFACTOR REGION Landscape? TITLE)      ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
    (ALLOW.BUTTON.EVENTS)
    (LET* ((STREAM (OPENPOSTSCRIPTSTREAM FILE (LIST 'DOCUMENT.NAME TITLE 'ROTATION Landscape?
                                                    'IMAGESIZEFACTOR SCALEFACTOR)))
           (IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (SCLIP (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA))
           SCALE)
          [COND
             [REGION (SETQ REGION (COPY REGION))                ; In case we need to change it.
                     [COND
                        ((< (fetch BITMAPWIDTH of BITMAP)
                            (+ (fetch (REGION LEFT) of REGION)
                               (fetch (REGION WIDTH) of REGION)))
                         (replace (REGION WIDTH) of REGION with (- (fetch BITMAPWIDTH of BITMAP)
                                                                   (fetch (REGION LEFT) of REGION]
                     (COND
                        ((< (fetch BITMAPHEIGHT of BITMAP)
                            (+ (fetch (REGION BOTTOM) of REGION)
                               (fetch (REGION HEIGHT) of REGION)))
                         (replace (REGION HEIGHT) of REGION with (- (fetch BITMAPHEIGHT of BITMAP)
                                                                    (fetch (REGION BOTTOM) of REGION]
             (T (SETQ REGION (create REGION
                                     LEFT _ 0
                                     BOTTOM _ 0
                                     WIDTH _ (fetch BITMAPWIDTH of BITMAP)
                                     HEIGHT _ (fetch BITMAPHEIGHT of BITMAP]
          (SETQ SCALE (TIMES POSTSCRIPT.BITMAP.SCALE (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA)))
          (BITBLT BITMAP (fetch (REGION LEFT) of REGION)
                 (fetch (REGION BOTTOM) of REGION)
                 STREAM
                 (PLUS (fetch (REGION LEFT) of SCLIP)
                       (QUOTIENT (DIFFERENCE (fetch (REGION WIDTH) of SCLIP)
                                             (TIMES SCALE (fetch (REGION WIDTH) of REGION)))
                                 2))
                 (PLUS (fetch (REGION BOTTOM) of SCLIP)
                       (QUOTIENT (DIFFERENCE (fetch (REGION HEIGHT) of SCLIP)
                                             (TIMES SCALE (fetch (REGION HEIGHT) of REGION)))
                                 2))
                 (fetch (REGION WIDTH) of REGION)
                 (fetch (REGION HEIGHT) of REGION)
                 'INPUT
                 'REPLACE)
          (CLOSEF STREAM)
          (FULLNAME STREAM])
```

(POSTSCRIPT.TEDIT

```
[LAMBDA (FILE PFILE)                                    ; Edited 18-Sep-91 18:16 by jds
```

;; Make a PS file from a TEdit document. If FILE is a string, make it into a symbol for the file-name. If it's a STREAM, use that stream.

```
[COND
   ((STRINGP FILE)
    (SETQ FILE (MKATOM FILE]
(SETQ FILE (OPENTEXTSTREAM FILE))
(TEDIT.FORMAT.HARDCOPY FILE PFILE T NIL NIL NIL 'POSTSCRIPT)
(CLOSEF? FILE)
PFILE])
```

(POSTSCRIPT.TEXT

```
[LAMBDA (FILE PSCFILE FONTS HEADING TABS)               ; Edited 23-Apr-89 11:31 by TAL
  (TEXTTOIMAGEFILE FILE PSCFILE 'POSTSCRIPT FONTS HEADING TABS `(REGION ,POSTSCRIPT.DEFAULT.PAGEREGION
                                                                 ROTATION ,(NOT (NOT
                                                                      POSTSCRIPT.TEXTFILE.LANDSCAPE
                                                                           ])
```

(POSTSCRIPTFILEP

```
[LAMBDA (FILE)                                          ; Edited 21-Nov-2023 17:04 by rmk
                                                        ; Edited  5-Mar-93 21:40 by rmk:
                                                        ; Edited 14-Jan-93 10:56 by jds
  (OR (CL:MEMBER (UNPACKFILENAME.STRING FILE 'EXTENSION)
            [CADR (ASSOC 'EXTENSION (CDR (ASSOC 'POSTSCRIPT PRINTFILETYPES]
            :TEST
            (FUNCTION STRING-EQUAL))
      (RESETLST
          [LET (STRM)
               [if (SETQ STRM (\GETSTREAM FILE 'INPUT T))
                   then (RESETSAVE (GETFILEPTR STRM)
                                   `(SETFILEPTR ,STRM OLDVALUE))
                           (SETFILEPTR STRM 0)
                 else (RESETSAVE (SETQ STRM (OPENSTREAM FILE 'INPUT))
                                 `(PROGN (CLOSEF OLDVALUE]
               (AND (EQ (BIN STRM)
                        (CHARCODE %%))
                    (EQ (BIN STRM)
                        (CHARCODE !])])
```

(MAKEEPSFILE

```
[LAMBDA (IMAGEOBJ FILENAME)                             ; Edited  7-Apr-94 14:48 by rmk:
```

;; Puts IMAGEOBJ on a 1-page encapsulated postscript file. The lower-left corner of the image box will be at 0,0 on the page.

```
(LET* [(STREAM (OPENIMAGESTREAM `{NODIRCORE}SCRATCH 'POSTSCRIPT))
       (IMAGEBOX (APPLY* (IMAGEOBJPROP IMAGEOBJ 'IMAGEBOXFN)
                         IMAGEOBJ STREAM))
       (BOUNDINGBOX (LIST 0 0 (FETCH XSIZE OF IMAGEBOX)
                          (FETCH YSIZE OF IMAGEBOX]
      [SETQ STREAM (OPENIMAGESTREAM FILENAME 'POSTSCRIPT `(BOUNDINGBOX (0 0 ,(FETCH XSIZE OF IMAGEBOX)
                                                                          ,(FETCH YSIZE OF IMAGEBOX]
      (MOVETO (FETCH XKERN OF IMAGEBOX)
              (FETCH YDESC OF IMAGEBOX)
             STREAM)
      (APPLY* (IMAGEOBJPROP IMAGEOBJ 'DISPLAYFN)
              IMAGEOBJ STREAM)
      (CLOSEF STREAM])

)

(DEFINEQ
```

(POSTSCRIPT.BITMAPSCALE

```
[LAMBDA (WIDTH HEIGHT)                                  ; Edited 29-Apr-98 08:46 by rmk:
                                                        ; Edited 20-Nov-92 14:52 by sybalsky:mv:envos
  (LET* ([PAGEREGION (\PS.SCALEREGION (/ 72 POSTSCRIPT.BITMAP.SCALE)
                          (CADR (FASSOC POSTSCRIPT.PAGETYPE POSTSCRIPT.PAGEREGIONS]
         (LONGEDGE (MAX (fetch (REGION WIDTH) of PAGEREGION)
                        (fetch (REGION HEIGHT) of PAGEREGION)))
         (SHORTEDGE (MIN (fetch (REGION WIDTH) of PAGEREGION)
                         (fetch (REGION HEIGHT) of PAGEREGION)))
         [MINDIMP (MIN (FQUOTIENT LONGEDGE (SETQ HEIGHT (TIMES HEIGHT POSTSCRIPT.BITMAP.SCALE)))
                       (FQUOTIENT SHORTEDGE (SETQ WIDTH (TIMES WIDTH POSTSCRIPT.BITMAP.SCALE]
         (MINDIML (MIN (FQUOTIENT SHORTEDGE HEIGHT)
                       (FQUOTIENT LONGEDGE WIDTH)))
         (PPL (if (EQ POSTSCRIPT.PREFER.LANDSCAPE 'ASK)
                  then (MENU \POSTSCRIPT.ORIENTATION.MENU)
                else POSTSCRIPT.PREFER.LANDSCAPE))
         MINDIM OTHERDIM SF1 SF2)
        (if PPL
            then (SETQ MINDIM MINDIML)
                 (SETQ OTHERDIM MINDIMP)
          else (SETQ MINDIM MINDIMP)
               (SETQ OTHERDIM MINDIML))
```

```
          (SETQ SF1 (if (GREATERP MINDIM 1)
                        then 1
                      elseif (GREATERP MINDIM 0.75)
                        then 0.75
                      elseif (GREATERP MINDIM 0.5)
                        then 0.5
                      elseif (GREATERP MINDIM 0.25)
                        then 0.25
                      else MINDIM))
          (SETQ SF2 (if (GREATERP OTHERDIM 1)
                        then 1
                      elseif (GREATERP OTHERDIM 0.75)
                        then 0.75
                      elseif (GREATERP OTHERDIM 0.5)
                        then 0.5
                      elseif (GREATERP OTHERDIM 0.25)
                        then 0.25
                      else OTHERDIM))
          (if (AND (LESSP SF1 1)
                   (LESSP SF1 SF2))
              then (CONS SF2 (NOT PPL))
            else (CONS SF1 PPL])
```

(POSTSCRIPT.CLOSESTRING

```
[LAMBDA (STREAM)                                          ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
       (COND
          ((fetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA)
           (POSTSCRIPT.OUTSTR STREAM ") ")
           (replace (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA with NIL)
           T)
          (T NIL])
```

(POSTSCRIPT.ENDPAGE

```
[LAMBDA (STREAM)                                          ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
       (POSTSCRIPT.SHOWACCUM STREAM)
       (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with NIL)
       (COND
          ((NOT (PROG1 (fetch (\POSTSCRIPTDATA POSTSCRIPTPAGEBLANK) of IMAGEDATA)
                       (POSTSCRIPT.PUTCOMMAND STREAM "grestore savepage restore ")))
           (POSTSCRIPT.PUTCOMMAND STREAM "showpage" :EOL)))
       ;; Force re-encoding of fonts, because the restore wipes out any you encoded while writing this page.
       (replace (\POSTSCRIPTDATA POSTSCRIPTFONTSUSED) of IMAGEDATA with NIL])
```

(POSTSCRIPT.OUTSTR

```
[LAMBDA (STREAM X)                                        ; Edited 14-Jul-89 14:05 by Matt Heffron
  (DECLARE (LOCALVARS . T))
  (COND
     ((FIXP X)                                            ; Common case, speed helps
      (\PS.BOUTFIXP STREAM X))
     [(STRINGP X)                                         ; Other common case
      (COND
         [(ffetch (STRINGP FATSTRINGP) of X)
          (for c infatstring X do (BOUT STREAM (\CHAR8CODE c]
         (T (\BOUTS STREAM (ffetch (STRINGP BASE) of X)
                   (ffetch (STRINGP OFFST) of X)
                   (ffetch (STRINGP LENGTH) of X]
     [(LITATOM X)
      (for c inatom X do (BOUT STREAM (\CHAR8CODE c]
     ((ZEROP X)
      (BOUT STREAM (CHARCODE 0)))
     (T [COND
           ((TYPEP X 'RATIO)
            (SETQ X (FLOAT X]
        (for c in (CHCON X) do (BOUT STREAM (\CHAR8CODE c])
```

(POSTSCRIPT.PUTBITMAPBYTES

```
[LAMBDA (STREAM BITMAP DELIMFLG)
  (DECLARE (GLOBALVARS PS.BITMAPARRAY)
           (LOCALVARS . T))                               ; Edited 12-Jun-2021 15:17 by rmk:
  (LET*
   ((WIDTH (fetch BITMAPWIDTH of BITMAP))
    (HEIGHT (fetch BITMAPHEIGHT of BITMAP))
    (BMBASE (fetch BITMAPBASE of BITMAP))
    (BYTESPERROW (LRSH (IPLUS WIDTH 7)
                       3))
    (BYTEOFFSETPERROW (LSH (fetch BITMAPRASTERWIDTH of BITMAP)
                           1))
    (PS.BITMAPARRAYBASE (fetch (ARRAYP BASE) of PS.BITMAPARRAY)))
   (COND
      (DELIMFLG (LET ((POS 0)
```

```
                    BYTE)
                   (BOUT STREAM (CHARCODE SPACE))
                   (BOUT STREAM (CHARCODE <))
                   (\BOUTEOL STREAM)
                   (for R from (SUB1 HEIGHT) to 0 by -1 as ROWOFFSET from (ITIMES (SUB1 HEIGHT)
                                                                                  BYTEOFFSETPERROW)
                      by (IMINUS BYTEOFFSETPERROW)
                      do (for B from 1 to BYTESPERROW as BYTEOFFSET from ROWOFFSET by 1
                            do (COND
                                   ((IGEQ POS 254)
                                    (\BOUTEOL STREAM)
                                    (SETQ POS 0)))
                               (SETQ BYTE (\GETBASEBYTE BMBASE BYTEOFFSET))
                               [BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 (LRSH BYTE 4]
                               (BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 BYTE)))
                               (SETQ POS (IPLUS POS 2)))
                         (\BOUTEOL STREAM)
                         (SETQ POS 0))
                   (BOUT STREAM (CHARCODE SPACE))
                   (BOUT STREAM (CHARCODE >))
                   (\BOUTEOL STREAM)))
        (T
         (LET*
          ((PRVBM (BITMAPCREATE WIDTH 1))
           (PRVBASE (fetch BITMAPBASE of PRVBM)))
          (for R from 0 to (SUB1 HEIGHT) as ROWOFFSET from (ITIMES (SUB1 HEIGHT)
                                                                   BYTEOFFSETPERROW)
             by (IMINUS BYTEOFFSETPERROW)
             do (LET ((POS 0)
                      (BYTEOFFSET ROWOFFSET)
                      (B 1)
                      (PRVO 0)
                      BYTE REPC)
                     [while (ILEQ B BYTESPERROW)
                        do (SETQ REPC
                                 (for BB from B to BYTESPERROW as BO from BYTEOFFSET by 1 as PO from PRVO by 1
                                    while (EQ (\GETBASEBYTE BMBASE BO)
                                              (\GETBASEBYTE PRVBASE PO))
                                    count T))
                           (COND
                              [(IGEQ REPC 3)
                               (SETQ B (IPLUS B REPC))
                               (SETQ BYTEOFFSET (IPLUS BYTEOFFSET REPC))
                               (SETQ PRVO (IPLUS PRVO REPC))
                               (while (CL:PLUSP (SETQ REPC (IDIFFERENCE REPC 1)))
                                  do (COND
                                         ((IGEQ POS 251)
                                          (\BOUTEOL STREAM)
                                          (SETQ POS 0)))
                                     (BOUT STREAM (CHARCODE B))
                                     (BOUT STREAM (CHARCODE 3))
                                     [COND
                                        ((IGEQ REPC 256)
                                         (BOUT STREAM (CHARCODE F))
                                         (BOUT STREAM (CHARCODE F)))
                                        (T [BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15
                                                                                                 (LRSH REPC 4]
                                           (BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 REPC]
                                     (SETQ REPC (IDIFFERENCE REPC 256))
                                     (SETQ POS (IPLUS POS 4]
                                (T (SETQ BYTE (\GETBASEBYTE BMBASE BYTEOFFSET))
                                   (SETQ REPC (for BB from B to BYTESPERROW as BO from BYTEOFFSET by 1
                                                 while (EQ (\GETBASEBYTE BMBASE BO)
                                                           BYTE)
                                                 count T))
                                   (COND
                                      [(IGEQ REPC 3)
                                       (SETQ B (IPLUS B REPC))
                                       (SETQ BYTEOFFSET (IPLUS BYTEOFFSET REPC))
                                       (SETQ PRVO (IPLUS PRVO REPC))
                                       (while (CL:PLUSP (SETQ REPC (IDIFFERENCE REPC 1)))
                                          do (COND
                                                 ((IGEQ POS 249)
                                                  (\BOUTEOL STREAM)
                                                  (SETQ POS 0)))
                                             (BOUT STREAM (CHARCODE B))
                                             (BOUT STREAM (CHARCODE 2))
                                             [COND
                                                ((IGEQ REPC 256)
                                                 (BOUT STREAM (CHARCODE F))
                                                 (BOUT STREAM (CHARCODE F)))
                                                (T [BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE
                                                                              (LOGAND 15 (LRSH REPC 4]
                                                   (BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 REPC]
                                             [BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15
                                                                                                   (LRSH BYTE 4]
                                             (BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 BYTE)))
```

```
                                        (SETQ REPC (IDIFFERENCE REPC 256))
                                        (SETQ POS (IPLUS POS 4]
                                  (T (SETQ BYTE (\GETBASEBYTE BMBASE BYTEOFFSET))
                                     (COND
                                        ((IGEQ POS 251)
                                         (\BOUTEOL STREAM)
                                         (SETQ POS 0)))
                                     [COND
                                        ((FMEMB BYTE '(178 179 180))
                                         ;; BYTE is B2, B3, or B4; quote it
                                         (BOUT STREAM (CHARCODE B))
                                         (BOUT STREAM (CHARCODE 4))
                                         (SETQ POS (IPLUS POS 2]
                                     [BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 (LRSH BYTE 4]
                                     (BOUT STREAM (\GETBASEBYTE PS.BITMAPARRAYBASE (LOGAND 15 BYTE)))
                                     (SETQ B (IPLUS B 1))
                                     (SETQ BYTEOFFSET (IPLUS BYTEOFFSET 1))
                                     (SETQ PRVO (IPLUS PRVO 1))
                                     (SETQ POS (IPLUS POS 2]
                          (\BOUTEOL STREAM))
                    (\MOVEBYTES BMBASE ROWOFFSET PRVBASE 0 BYTESPERROW])
```

(**POSTSCRIPT.PUTCOMMAND**

```
[LAMBDA S.STRS                                                    ; Edited 12-Jun-2021 15:14 by rmk:
  (LET* ((STREAM (ARG S.STRS 1))
         (IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         S#S)
        (freplace (\POSTSCRIPTDATA POSTSCRIPTPAGEBLANK) of IMAGEDATA with NIL)
        (COND
           ((ffetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA)
            (POSTSCRIPT.SHOWACCUM STREAM)))
        (COND
           ((ffetch (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA)
            (\SETXFORM.PSC STREAM IMAGEDATA)))
        (for STR# from 2 to S.STRS do (COND
                                         ((EQ (SETQ S#S (ARG S.STRS STR#))
                                              :EOL)
                                          (\BOUTEOL STREAM))
                                         (T (POSTSCRIPT.OUTSTR STREAM S#S])
```

(**POSTSCRIPT.SET-FAKE-LANDSCAPE**

```
[LAMBDA (STREAM ROTATION)                                         ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
```

;; Set up for (or disable) fake landscaping

;; we only know 90 degrees of rotation for now (0 means portrait, anything else is landscape).

```
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OLAND (COND
                   ((fetch (\POSTSCRIPTDATA POSTSCRIPTLANDSCAPE) of IMAGEDATA)
                    90)
                   (T 0)))
         LAND C0 P0 C P ML MB MR MT)
        (COND
           ((AND ROTATION (NEQ (SETQ LAND (NOT (ZEROP ROTATION)))
                               (fetch (\POSTSCRIPTDATA POSTSCRIPTLANDSCAPE) of IMAGEDATA)))
            (POSTSCRIPT.SHOWACCUM STREAM)
            (\DSPTRANSLATE.PSC STREAM 0 0)
            (SETQ C0 (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA))
            (SETQ P0 (fetch (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA))
            (SETQ C (create REGION
                            WIDTH _ (fetch (REGION HEIGHT) of C0)
                            HEIGHT _ (fetch (REGION WIDTH) of C0)))
            (SETQ P (create REGION
                            LEFT _ 0
                            BOTTOM _ 0
                            WIDTH _ (fetch (REGION HEIGHT) of P0)
                            HEIGHT _ (fetch (REGION WIDTH) of P0)))
            [COND
               (LAND (replace (REGION LEFT) of C with (fetch (REGION BOTTOM) of C0))
                     [replace (REGION BOTTOM) of C with (- (fetch (REGION WIDTH) of P0)
                                                           (+ (fetch (REGION LEFT) of C0)
                                                              (fetch (REGION WIDTH) of C0]
                     (SETQ ML (fetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA))
                     (SETQ MB (- (fetch (REGION WIDTH) of P0)
                                 (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA)
                                 1))
                     (SETQ MR (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA))
                     (SETQ MT (- (fetch (REGION WIDTH) of P0)
                                 (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
                                 1)))
               (T [replace (REGION LEFT) of C with (- (fetch (REGION HEIGHT) of P0)
                                                      (+ (fetch (REGION BOTTOM) of C0)
                                                         (fetch (REGION HEIGHT) of C0]
                  (replace (REGION BOTTOM) of C with (fetch (REGION LEFT) of C0))
                  (SETQ ML (- (fetch (REGION HEIGHT) of P0)
```

```
                              (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA)
                              1))
                    (SETQ MB (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA))
                    (SETQ MR (- (fetch (REGION HEIGHT) of P0)
                              (fetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA)
                              1))
                    (SETQ MT (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA]
               (replace (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA with C)
               (replace (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA with P)
               (replace (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA with ML)
               (replace (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA with MB)
               (replace (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA with MR)
               (replace (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA with MT)
               (replace (\POSTSCRIPTDATA POSTSCRIPTLANDSCAPE) of IMAGEDATA with LAND)
               (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with T)
               (\DSPRESET.PSC STREAM)))
          OLAND])
```

(POSTSCRIPT.SHOWACCUM

```
[LAMBDA (STREAM)                                                    ; Edited 12-Jun-2021 15:16 by rmk:
```

;; Send commands to SHOW the accumulated characters. Uses S (= SHOW) for regular characters.

;; Uses WIDTHSHOW if the space-factor isn't 1

;; Uses ASHOW if a KERN value is on STREAM's properties

;; USES AWIDTHSHOW if both space-factor != 1 and there's a KERN value.

```
(LET ((IMAGEDATA (ffetch (STREAM IMAGEDATA) of STREAM))
      KERN)
     (COND
        ((fetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA)
         (SETQ KERN (STREAMPROP STREAM 'KERN))
         [COND
            [(EQP (ffetch (\POSTSCRIPTDATA POSTSCRIPTSPACEFACTOR) of IMAGEDATA)
                  1)
             (COND
                (KERN (POSTSCRIPT.OUTSTR STREAM (CONCAT ") " KERN " 0 3 -1 roll ashow")))
                (T (POSTSCRIPT.OUTSTR STREAM ") S"]
            (T (POSTSCRIPT.OUTSTR STREAM ") ")
               (POSTSCRIPT.OUTSTR STREAM (DIFFERENCE (ffetch (\POSTSCRIPTDATA POSTSCRIPTSPACEWIDTH)
                                                             of IMAGEDATA)
                                                     (ffetch (\POSTSCRIPTDATA POSTSCRIPTNATURALSPACEWIDTH)
                                                        of IMAGEDATA)))
               (COND
                  (KERN (POSTSCRIPT.OUTSTR STREAM (CONCAT " 0 " (CHARCODE SPACE)
                                                          " " KERN " 0 " " 6 -1 roll awidthshow")))
                  (T (POSTSCRIPT.OUTSTR STREAM (CONSTANT (CONCAT " 0 " (CHARCODE SPACE)
                                                                 " 4 -1 roll widthshow"]
         (\BOUTEOL STREAM)
         (freplace (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA with NIL])
```

(POSTSCRIPT.STARTPAGE

```
[LAMBDA (STREAM)                                                    ; Edited 12-Jun-2021 14:52 by rmk:
```

;; Start up a new page in a Postscript document.

```
(LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
      NEW-PAGE)
     (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with NIL)
                                                                    ; shouldnt need this
     (SETQ NEW-PAGE (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTPAGENUM) of IMAGEDATA)))
                                                                    ; Page number goes up by 1
```

;; Print the "Document structuring" info for the page, then the initial page setup

```
     (POSTSCRIPT.PUTCOMMAND STREAM :EOL "%%%%Page: " NEW-PAGE " " NEW-PAGE :EOL "%%%%BeginPageSetup" :EOL
            "/savepage save def" :EOL (FQUOTIENT 1 \PS.SCALE0)
            " imagesizefactor mul dup scale" :EOL "%%%%EndPageSetup" :EOL)
     (\SETXFORM.PSC STREAM IMAGEDATA T)
```

;; Lisp depends on the current font being carried over from page to page, but in postscript there is no current font at the beginning of a page,
;; so force a setfont.

```
     (replace (\POSTSCRIPTDATA POSTSCRIPTFONTCHANGEDFLG) of IMAGEDATA with T)
     (replace (\POSTSCRIPTDATA POSTSCRIPTPAGEBLANK) of IMAGEDATA with T)
                                                                    ; nothing printed yet...
     (COND
        ((fetch (\POSTSCRIPTDATA POSTSCRIPTHEADING) of IMAGEDATA)
```

;; Here we handle headings.

```
         (LET [(FONT (\DSPFONT.PSC STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTHEADINGFONT) of IMAGEDATA]
               (\DSPRESET.PSC STREAM)
               (POSTSCRIPT.OUTSTR STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTHEADING) of IMAGEDATA))
               (RELMOVETO (CONSTANT (TIMES 72 \PS.SCALE0))
                     0 STREAM)                                      ; Skip an inch before page number
               (POSTSCRIPT.OUTSTR STREAM "Page ")
               (POSTSCRIPT.OUTSTR STREAM NEW-PAGE)
               (\TERPRI.PSC STREAM)                                 ; Skip 2 lines
               (\TERPRI.PSC STREAM)
```

```
                (\DSPFONT.PSC STREAM FONT)))
          (T (\DSPRESET.PSC STREAM])
```

(\POSTSCRIPTTAB

```
[LAMBDA (POSTSCRIPTDATA)                                    ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
  (LET [(TABSPACE (TIMES 8 (ffetch FONTAVGCHARWIDTH of (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of POSTSCRIPTDATA]
       (IDIFFERENCE TABSPACE (IREMAINDER (IDIFFERENCE (ffetch (\POSTSCRIPTDATA POSTSCRIPTX) of POSTSCRIPTDATA)
                                                      (ffetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of POSTSCRIPTDATA
                                                              ))
                                         TABSPACE])
```

(\PS.BOUTFIXP

```
[LAMBDA (STREAM N)                                          ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
```

;; BOUT the decimal representation of N to STREAM using temp storage from the imagedata. Done this way for speed.

```
  (DECLARE (LOCALVARS . T))
  [COND
     ((MINUSP N)
      (BOUT STREAM (CHARCODE -))
      (SETQ N (IMINUS N]
  (COND
     [(LESSP N 10)
      (BOUT STREAM (IPLUS N (CHARCODE 0]
     [(LESSP N 1000000000)
      (LET ([BASE (fetch (ARRAYP BASE) of (fetch (\POSTSCRIPTDATA POSTSCRIPTTEMPARRAY)
                                             of (fetch (STREAM IMAGEDATA) of STREAM]
            (i (SUB1 \PS.TEMPARRAYLEN)))
           [for old i by -1 do (\PUTBASEBYTE BASE i (IPLUS (IREMAINDER N 10)
                                                           (CHARCODE 0)))
              repeatwhile (NEQ 0 (SETQ N (IQUOTIENT N 10]
           (\BOUTS STREAM BASE i (IDIFFERENCE \PS.TEMPARRAYLEN i]
     (T                                                     ; Just in case we get a bignum
        (for c in (CHCON N) do (BOUT STREAM (\CHAR8CODE c])
```

(\PS.SCALEHACK

```
[LAMBDA (STREAM SCALEFACTOR)                                ; Edited 20-Nov-92 15:11 by sybalsky:mv:envos
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OLDSCALE (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALEHACK) of IMAGEDATA))
         FACTOR)
        (COND
           ((AND (NUMBERP SCALEFACTOR)
                 (NOT (EQP OLDSCALE SCALEFACTOR)))
            (POSTSCRIPT.SHOWACCUM STREAM)
            (SETQ FACTOR (/ OLDSCALE SCALEFACTOR))
            [for REG in (LIST (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA)
                              (fetch (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA))
               do (change (fetch (REGION LEFT) of REG)
                          (FIXR (CL:* DATUM FACTOR)))
                  (change (fetch (REGION BOTTOM) of REG)
                          (FIXR (CL:* DATUM FACTOR)))
                  (change (fetch (REGION WIDTH) of REG)
                          (FIXR (CL:* DATUM FACTOR)))
                  (change (fetch (REGION HEIGHT) of REG)
                          (FIXR (CL:* DATUM FACTOR]
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (change (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA)
                    (FIXR (CL:* DATUM FACTOR)))
            (replace (\POSTSCRIPTDATA POSTSCRIPTSCALEHACK) of IMAGEDATA with SCALEFACTOR)
            (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with T)))
        OLDSCALE])
```

(\PS.SCALEREGION

```
[LAMBDA (SCALE REGION)                                      ; Edited  5-Apr-89 16:15 by TAL
                                                            ; Scales a region
  (create REGION
          LEFT _ (FIXR (TIMES SCALE (fetch (REGION LEFT) of REGION)))
          BOTTOM _ (FIXR (TIMES SCALE (fetch (REGION BOTTOM) of REGION)))
          WIDTH _ (FIXR (TIMES SCALE (fetch (REGION WIDTH) of REGION)))
          HEIGHT _ (FIXR (TIMES SCALE (fetch (REGION HEIGHT) of REGION])
```

(\SCALEDBITBLT.PSC

```
[LAMBDA (SOURCEBITMAP SOURCELEFT SOURCEBOTTOM STREAM DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT SOURCETYPE
               OPERATION TEXTURE CLIPPINGREGION CLIPPEDSOURCELEFT CLIPPEDSOURCEBOTTOM SCALE)
                                                                                ; Edited  8-May-2018 19:33 by rmk:
                                                                                ; Edited  8-May-2018 15:05 by rmk:
                                                                                ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos

    ;; Postscript can only handle OPERATION REPLACE and PAINT.  SOURCETYPE = TEXTURE is converted to BLTSHADE before getting here (so
    ;; the TEXTURE argument can be ignored).  If the destination region lies completely outside the clipping region we do nothing, otherwise we output
    ;; the whole thing and let the printer clip.  Could be more clever.

    (OR (NUMBERP SCALE)
        (SETQ SCALE 1))
    (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (SCALE1 (TIMES SCALE (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA)))
           (SCALE2 (TIMES SCALE1 (OR (NUMBERP POSTSCRIPT.BITMAP.SCALE)
                                     1)))
           DESTREGION
           (BITMAPWIDTH (fetch BITMAPWIDTH of SOURCEBITMAP))
           (BITMAPHEIGHT (fetch BITMAPHEIGHT of SOURCEBITMAP))
           TEMPBM)
          [COND
             ((NULL DESTINATIONLEFT)
              (SETQ DESTINATIONLEFT (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA]
          [COND
             ((NULL DESTINATIONBOTTOM)
              (SETQ DESTINATIONBOTTOM (fetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA]
          (COND
             ((OR (NULL WIDTH)
                  (NULL HEIGHT))
              (SETQ WIDTH BITMAPWIDTH)
              (SETQ HEIGHT BITMAPHEIGHT)))
          (COND
             ((GREATERP WIDTH BITMAPWIDTH)
              (SETQ WIDTH BITMAPWIDTH)))
          (COND
             ((GREATERP HEIGHT BITMAPHEIGHT)
              (SETQ HEIGHT BITMAPHEIGHT)))
          [SETQ DESTREGION (INTERSECTREGIONS (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA)
                                   (CREATEREGION DESTINATIONLEFT DESTINATIONBOTTOM (TIMES SCALE1 WIDTH)
                                          (TIMES SCALE1 HEIGHT]
          (COND
             ((AND DESTREGION (OR (NULL CLIPPINGREGION)
                                  (REGIONSINTERSECTP DESTREGION CLIPPINGREGION)))
              [COND
                 ((AND (EQ SOURCELEFT 0)
                       (EQ SOURCEBOTTOM 0)
                       (EQP WIDTH BITMAPWIDTH)
                       (EQP HEIGHT BITMAPHEIGHT))                    ; Avoid copy if sending entire bitmap
                  (SETQ TEMPBM SOURCEBITMAP))
                 (T (SETQ TEMPBM (BITMAPCREATE WIDTH HEIGHT 1))
                    (BITBLT SOURCEBITMAP SOURCELEFT SOURCEBOTTOM TEMPBM 0 0 WIDTH HEIGHT SOURCETYPE 'REPLACE]
              (POSTSCRIPT.PUTCOMMAND STREAM "/bitbltsave save def " DESTINATIONLEFT " " DESTINATIONBOTTOM "
                     translate " (TIMES SCALE2 WIDTH)
                     " "
                     (TIMES SCALE2 HEIGHT)
                     " scale " WIDTH " " HEIGHT (COND
                                                   ((EQ OPERATION 'PAINT)
                                                    " true")
                                                   (T ;; RMK: For REPLACE, was "false", but then white was black.
                                                      " true"))
                     " thebitimage" :EOL)
              (POSTSCRIPT.PUTBITMAPBYTES STREAM TEMPBM NIL)
              (POSTSCRIPT.PUTCOMMAND STREAM :EOL "bitbltsave restore" :EOL)
              (\MOVETO.PSC STREAM DESTINATIONLEFT DESTINATIONBOTTOM)
              T)
             (T NIL])
```

(\SETPOS.PSC

```
[LAMBDA (STREAM IMAGEDATA)                                      ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (POSTSCRIPT.PUTCOMMAND STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
         " "
         (ffetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA)
         " M ")
  (freplace (\POSTSCRIPTDATA POSTSCRIPTMOVEFLG) of IMAGEDATA with NIL])
```

(\SETXFORM.PSC

```
[LAMBDA (STREAM IMAGEDATA NORESTORE)                            ; Edited 28-Dec-94 17:59 by jds

    ;; Write transforms into the PS file to make what it prints match what we think it should print.

    (LET ((CLIP (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA)))
         (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with NIL)
         (COND
            ((NOT NORESTORE)
```

```
        (POSTSCRIPT.OUTSTR STREAM "grestore ")))
    (POSTSCRIPT.PUTCOMMAND STREAM "gsave" :EOL)
```

;; Scaling

```
(COND
   ((NOT (EQP (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALEHACK) of IMAGEDATA)
              1))
    (POSTSCRIPT.PUTCOMMAND STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALEHACK) of IMAGEDATA)
           " dup scale" :EOL)))
```

;; Landscape mode (as in POSTSCRIPT.PREFER.LANDSCAPE, not as in TEdit doing landscaping)

```
(COND
   ((fetch (\POSTSCRIPTDATA POSTSCRIPTLANDSCAPE) of IMAGEDATA)
    (POSTSCRIPT.OUTSTR STREAM " 90 rotate 0 -61200 imagesizefactor div translate ")))
```

;; Any rotation that is in effect.

```
(POSTSCRIPT.PUTCOMMAND STREAM " " (fetch (\POSTSCRIPTDATA POSTSCRIPTROTATION) of IMAGEDATA)
       " rotate " :EOL)
```

;; Any translations that are in effect.

```
(COND
   ([NOT (AND (ZEROP (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA))
              (ZEROP (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA]
    (POSTSCRIPT.PUTCOMMAND STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA)
           " "
           (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA)
           " translate" :EOL)))
```

;; Clipping region:

```
(POSTSCRIPT.PUTCOMMAND STREAM " " (fetch (REGION HEIGHT) of CLIP)
       " "
       (fetch (REGION WIDTH) of CLIP)
       " "
       (fetch (REGION LEFT) of CLIP)
       " "
       (fetch (REGION BOTTOM) of CLIP)
       " CLP" :EOL)
```

;; And force recaching of location and font.

```
(replace (\POSTSCRIPTDATA POSTSCRIPTMOVEFLG) of IMAGEDATA with T)
(replace (\POSTSCRIPTDATA POSTSCRIPTFONTCHANGEDFLG) of IMAGEDATA with T])
```

(\STRINGWIDTH.PSC

```
[LAMBDA (STREAM STR RDTBL)                                  ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (ffetch (STREAM IMAGEDATA) of STREAM)))
       (\STRINGWIDTH.GENERIC STR (fetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of IMAGEDATA)
              RDTBL
              (ffetch (\POSTSCRIPTDATA POSTSCRIPTSPACEWIDTH) of IMAGEDATA])
```

(\SWITCHFONTS.PSC

```
[LAMBDA (STREAM POSTSCRIPTDATA)                             ; Edited 23-May-93 12:04 by rmk:
                                                            ; Edited 11-May-93 02:11 by jds
```

;; Actually emit the PS commands to change the font. If the new font hasn't been used (on this page) before, re-encode it to support accented ;; characters.

```
(LET* [(FONT (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of POSTSCRIPTDATA))
       (FONTID (fetch (PSCFONT IL-FONTID) of (LISTGET (fetch (FONTDESCRIPTOR OTHERDEVICEFONTPROPS)
                                                             of FONT)
                                                      'PSCFONT]
      [COND
         [(LISTP FONTID)
          [COND
             ((MEMB (fetch (FONTID FONTIDNAME) of FONTID)
                    (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONTSUSED) of POSTSCRIPTDATA)))
             ((MEMB (fetch (FONTID FONTIDNAME) of FONTID)
                    *POSTSCRIPT-UNACCENTED-FONTS*))
             (T ;; This font hasn't been used on this page yet.  Re-encode it to include accented characters.
                (POSTSCRIPT.PUTCOMMAND STREAM "/" (fetch (FONTID FONTIDNAME) of FONTID)
                       " /"
                       (CONCAT (fetch (FONTID FONTIDNAME) of FONTID)
                              "-Acnt")
                       " encodefont" :EOL)
                (CL:PUSH (fetch (FONTID FONTIDNAME) of FONTID)
                       (FFETCH (\POSTSCRIPTDATA POSTSCRIPTFONTSUSED) OF POSTSCRIPTDATA]
          (COND
             ((MEMB (fetch (FONTID FONTIDNAME) of FONTID)
                    *POSTSCRIPT-UNACCENTED-FONTS*)
              (FREPLACE (\POSTSCRIPTDATA POSTSCRIPTACCENTED) OF POSTSCRIPTDATA WITH NIL)
              (POSTSCRIPT.PUTCOMMAND STREAM "/" (fetch (FONTID FONTIDNAME) of FONTID)
                     " findfont ["
                     (TIMES (fetch (FONTID FONTXFACTOR) of FONTID)
                            (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                            100)
                     " 0 "
```

```
                        (TIMES (fetch (FONTID FONTOBLIQUEFACTOR) of FONTID)
                               (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                               100)
                        " "
                        (TIMES (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                               100)
                        " 0 0] makefont setfont" :EOL))
                 (T (FREPLACE (\POSTSCRIPTDATA POSTSCRIPTACCENTED) OF POSTSCRIPTDATA WITH T)
                    (POSTSCRIPT.PUTCOMMAND STREAM "/" (CONCAT (fetch (FONTID FONTIDNAME) of FONTID)
                                                              "-Acnt")
                           " findfont ["
                           (TIMES (fetch (FONTID FONTXFACTOR) of FONTID)
                                  (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                                  100)
                           " 0 "
                           (TIMES (fetch (FONTID FONTOBLIQUEFACTOR) of FONTID)
                                  (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                                  100)
                           " "
                           (TIMES (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                                  100)
                           " 0 0] makefont setfont" :EOL]
              (T [COND
                    ((MEMB FONTID (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONTSUSED) of POSTSCRIPTDATA)))
                    ((MEMB FONTID *POSTSCRIPT-UNACCENTED-FONTS*))
                    (T ;; This font hasn't been used on this page yet.  Re-encode it to include accented characters.
                       (POSTSCRIPT.PUTCOMMAND STREAM "/" FONTID " /" (CONCAT FONTID "-Acnt")
                              " encodefont" :EOL)
                       (CL:PUSH FONTID (FFETCH (\POSTSCRIPTDATA POSTSCRIPTFONTSUSED) OF POSTSCRIPTDATA]
                 (COND
                    ((MEMB FONTID *POSTSCRIPT-UNACCENTED-FONTS*)
                     (freplace (\POSTSCRIPTDATA POSTSCRIPTACCENTED) of POSTSCRIPTDATA with NIL)
                     (POSTSCRIPT.PUTCOMMAND STREAM (TIMES (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                                                          100)
                            " /" FONTID " F" :EOL))
                    (T (freplace (\POSTSCRIPTDATA POSTSCRIPTACCENTED) of POSTSCRIPTDATA with T)
                       (POSTSCRIPT.PUTCOMMAND STREAM (TIMES (fetch (FONTDESCRIPTOR FONTSIZE) of FONT)
                                                            100)
                              " /"
                              (CONCAT FONTID "-Acnt")
                              " F" :EOL]
           (replace (\POSTSCRIPTDATA POSTSCRIPTFONTCHANGEDFLG) of POSTSCRIPTDATA with NIL])
```

(\TERPRI.PSC

```
  [LAMBDA (STREAM)                                          ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
    (LET* [(IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (NEWY (PLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA)
                       (ffetch (\POSTSCRIPTDATA POSTSCRIPTLINESPACING) of IMAGEDATA]
          (COND
             ([LESSP NEWY (IPLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA)
                                 (fetch (FONTDESCRIPTOR \SFDescent) of (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONT)
                                                                               of IMAGEDATA]
              (DSPNEWPAGE STREAM))
             (T (replace (STREAM CHARPOSITION) of STREAM with 0)
                (\MOVETO.PSC STREAM (ffetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
                       NEWY)))
          NIL])

)

;; DIG operations:

(DEFINEQ
```

(\BITBLT.PSC

```
  [LAMBDA (SOURCEBITMAP SOURCELEFT SOURCEBOTTOM STREAM DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT SOURCETYPE
                 OPERATION TEXTURE CLIPPINGREGION CLIPPEDSOURCELEFT CLIPPEDSOURCEBOTTOM)
                                                            ; Edited  7-Apr-89 19:53 by TAL
    (\SCALEDBITBLT.PSC SOURCEBITMAP SOURCELEFT SOURCEBOTTOM STREAM DESTINATIONLEFT DESTINATIONBOTTOM WIDTH
          HEIGHT SOURCETYPE OPERATION TEXTURE CLIPPINGREGION CLIPPEDSOURCELEFT CLIPPEDSOURCEBOTTOM 1])
```

(\BLTSHADE.PSC

```
  [LAMBDA (TEXTURE STREAM DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT OPERATION CLIPPINGREGION)
                                                            ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
    ;; Maybe we should do something with OPERATION
    (LET ((RGN (CREATEREGION DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT))
          (IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
          TEXTUREBM TEXTUREWIDTH LEFT BOTTOM WIDTH HEIGHT)
         [COND
            [CLIPPINGREGION (SETQ RGN (INTERSECTREGIONS RGN CLIPPINGREGION (fetch (\POSTSCRIPTDATA
                                                                                   POSTSCRIPTCLIPPINGREGION
                                                                                   )
                                                                              of IMAGEDATA]
```

```
      (T (SETQ RGN (INTERSECTREGIONS RGN (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA]
   (COND
      (RGN (SETQ LEFT (fetch (REGION LEFT) of RGN))
           (SETQ BOTTOM (fetch (REGION BOTTOM) of RGN))
           (SETQ WIDTH (CL:1- (fetch (REGION WIDTH) of RGN)))
           (SETQ HEIGHT (CL:1- (fetch (REGION HEIGHT) of RGN)))
           [COND
              ((FIXP TEXTURE)
               (SETQ TEXTURE (SELECT TEXTURE ((BLACKSHADE -1)
                                              0.0)
                                    (WHITESHADE 1.0)
                                    TEXTURE]
           [COND
              ((AND (FLOATP TEXTURE)
                    (<= 0.0 TEXTURE 1.0))
               (POSTSCRIPT.PUTCOMMAND STREAM HEIGHT " " WIDTH " " LEFT " " BOTTOM " " TEXTURE " R" :EOL)
               )
              ((OR (TEXTUREP TEXTURE)
                   (NULL TEXTURE))
               (SETQ TEXTUREBM (BITMAPCREATE 16 16 1))
               (SETQ TEXTUREWIDTH 16)
               (BLTSHADE TEXTURE TEXTUREBM))
              ((BITMAPP TEXTURE)
               (SETQ TEXTUREWIDTH (MIN (fetch BITMAPWIDTH of TEXTUREBM)
                                       (fetch BITMAPHEIGHT of TEXTUREBM)))
               (SETQ TEXTUREBM (BITMAPCREATE TEXTUREWIDTH TEXTUREWIDTH 1))
               (BITBLT TEXTURE 0 0 TEXTUREBM 0 0 TEXTUREWIDTH TEXTUREWIDTH 'INPUT 'REPLACE]
           (COND
              (TEXTUREBM (POSTSCRIPT.PUTCOMMAND STREAM "gsave newpath ")
                     (POSTSCRIPT.PUTCOMMAND STREAM "100 100 scale " (QUOTIENT LEFT 100.0)
                            " "
                            (QUOTIENT BOTTOM 100.0)
                            " M "
                            (SETQ WIDTH (QUOTIENT WIDTH 100.0))
                            " 0 rlineto 0 "
                            (QUOTIENT HEIGHT 100.0)
                            " rlineto "
                            (MINUS WIDTH)
                            " 0 rlineto closepath" :EOL)
                     (POSTSCRIPT.PUTBITMAPBYTES STREAM TEXTUREBM T)
                     (POSTSCRIPT.PUTCOMMAND STREAM TEXTUREWIDTH " " (LSH (fetch BITMAPRASTERWIDTH
                                                                            of TEXTUREBM)
                                                                         1)
                            " 0 "
                            (TIMES 72 (QUOTIENT (DSPSCALE NIL STREAM)
                                                100.0))
                            " findresolution " TEXTUREWIDTH " div div ceiling " POSTSCRIPT.TEXTURE.SCALE
                            " mul setpattern eofill" :EOL "grestore" :EOL)))
              (\MOVETO.PSC STREAM DESTINATIONLEFT DESTINATIONBOTTOM)
              T)
      (T NIL])
```

(\CHARWIDTH.PSC

```
[LAMBDA (STREAM CHARCODE)                                   ; Edited  8-May-93 11:19 by rmk:
  (COND
     ((EQ CHARCODE (CHARCODE SPACE))
      (fetch (\POSTSCRIPTDATA POSTSCRIPTSPACEWIDTH) of (ffetch (STREAM IMAGEDATA) of STREAM)))
     ((\FGETCHARWIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of (ffetch (STREAM IMAGEDATA) of STREAM))
            CHARCODE])
```

(\CREATECHARSET.PSC

```
[LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET FONTDESC NOSLUG?)
                                                            ; Edited  8-May-93 22:55 by rmk:
  (LET* ((CSINFO (CREATE CHARSETINFO
                         OFFSETS _ NIL))
         (WIDTHS (FETCH (CHARSETINFO WIDTHS) OF CSINFO)))
        (REPLACE (CHARSETINFO IMAGEWIDTHS) OF CSINFO WITH WIDTHS)

        ;; Make imagewidths point to widths.  Shouldn't matter to anyone, since imagewidths really has to do with bitmaps etc.  But...

        (CL:UNLESS (EQ CHARSET 0)

            ;; For all charsets other than 0, initialize widths with width of black box=average char width.  We know that the AVGCHARWIDTH field
            ;; of the FONTDESC will eventually be the width of A, but that might not be filled in when this is executed inside
            ;; POSTSCRIPT.FONTCREATE--it's only after the return to FONTCREATE itself that this gets filled in.  However, we do know that
            ;; charset 0 is all set up before any other characters are dealt with.

            (FOR I (AVGCHARWIDTH _ (CHARWIDTH (CHARCODE A)
                                              FONTDESC))
               FROM 0 TO 255 FIRST (CL:WHEN (EQ 0 AVGCHARWIDTH)

                                        ;; This is what \AVGCHARWIDTH in FONT does, but we don't have it here.  Just to be
                                        ;; extremely safe.

                                        [SETQ AVGCHARWIDTH (MAX 1 (FIXR (FTIMES 0.6 (FONTPROP FONTDESC
                                                                                              'HEIGHT])
               DO (\FSETWIDTH WIDTHS I AVGCHARWIDTH)))
        CSINFO])
```

(\DRAWARC.PSC

```
[LAMBDA (STREAM CENTERX CENTERY RADIUS STARTANGLE NDEGREES BRUSH DASHING)
                                                        ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        WIDTH COLOR)
       [COND
          ((NUMBERP BRUSH)
           (SETQ WIDTH BRUSH))
          ((LISTP BRUSH)
           (COND
              ((NEQ (fetch BRUSHSHAPE of BRUSH)
                    'ROUND)
               (printout T T "[In \DRAWARC.PSC: Non-ROUND BRUSH not supported.]
                      [Using ROUND BRUSH]" T)))
           (SETQ WIDTH (fetch BRUSHSIZE of BRUSH))
           (SETQ COLOR (fetch BRUSHCOLOR of BRUSH)))
          (T                                            ; If FUNCTIONAL BRUSH big trouble!
             (printout T T "[In \DRAWARC.PSC: Functional BRUSH not supported.]
                    [Using ROUND 1 point BRUSH]" T)
             (SETQ WIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA]
       (COND
          ((NOT (ZEROP WIDTH))
           (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
           (COND
              ((FLOATP COLOR)
               (POSTSCRIPT.PUTCOMMAND STREAM COLOR " setgray ")
                                                        ; COLOR is specified in POSTSCRIPT setgray notation.

               ))
           (COND
              ((LISTP DASHING)
               (POSTSCRIPT.OUTSTR STREAM " [")
               (for D in DASHING do (POSTSCRIPT.PUTCOMMAND STREAM (TIMES D WIDTH)
                                           " "))
               (POSTSCRIPT.PUTCOMMAND STREAM "] 0 setdash" :EOL)
                                                        ; Since Interlisp DASHING are in terms of BRUSH units, we must
                                                        ; multiply by the brush size.

               ))
           (POSTSCRIPT.PUTCOMMAND STREAM WIDTH " setlinewidth 1 setlinecap 1 setlinejoin " CENTERX " "
                  CENTERY " " RADIUS " " STARTANGLE " " (+ STARTANGLE NDEGREES)
                  " arc stroke" :EOL "grestore" :EOL)))
       (\MOVETO.PSC STREAM CENTERX CENTERY])
```

(\DRAWCIRCLE.PSC

```
[LAMBDA (STREAM CENTERX CENTERY RADIUS BRUSH DASHING)          ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        WIDTH COLOR)
       [COND
          ((NUMBERP BRUSH)
           (SETQ WIDTH BRUSH))
          ((LISTP BRUSH)
           (COND
              ((NEQ (fetch BRUSHSHAPE of BRUSH)
                    'ROUND)
               (printout T T "[In \DRAWCIRCLE.PSC: Non-ROUND BRUSH not supported.]
                      [Using ROUND BRUSH]" T)))
           (SETQ WIDTH (fetch BRUSHSIZE of BRUSH))
           (SETQ COLOR (fetch BRUSHCOLOR of BRUSH)))
          (T                                            ; If FUNCTIONAL BRUSH big trouble!
             (printout T T "[In \DRAWCIRCLE.PSC: Functional BRUSH not supported.]
                    [Using (ROUND 1) BRUSH]" T)
             (SETQ WIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA]
       (COND
          ((NOT (ZEROP WIDTH))
           (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
           (COND
              ((FLOATP COLOR)
               (POSTSCRIPT.PUTCOMMAND STREAM COLOR " setgray ")
                                                        ; COLOR is specified in POSTSCRIPT setgray notation.

               ))
           (COND
              ((LISTP DASHING)
               (POSTSCRIPT.OUTSTR STREAM " [")
               (for D in DASHING do (POSTSCRIPT.PUTCOMMAND STREAM (TIMES D WIDTH)
                                           " "))
               (POSTSCRIPT.PUTCOMMAND STREAM "] 0 setdash" :EOL)
                                                        ; Since Interlisp DASHING are in terms of BRUSH units, we must
                                                        ; multiply by the brush size.

               ))
           (POSTSCRIPT.PUTCOMMAND STREAM WIDTH " setlinewidth 1 setlinecap 1 setlinejoin " CENTERX " "
                  CENTERY " " RADIUS " 0 360 arc stroke" :EOL "grestore" :EOL)))
```

```
        (\MOVETO.PSC STREAM CENTERX CENTERY])
```

(\DRAWCURVE.PSC

```
[LAMBDA (STREAM KNOTS CLOSED BRUSH DASHING)                    ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        WIDTH SHAPE COLOR PSPLINE XA YA DXA DYA N PREVX PREVY PREV-DX3 PREV-DY3)
       [COND
          ((NUMBERP BRUSH)
           (SETQ WIDTH BRUSH)
           (SETQ SHAPE 'ROUND))
          ((LISTP BRUSH)
           (SETQ WIDTH (fetch BRUSHSIZE of BRUSH))
           (SETQ SHAPE (fetch BRUSHSHAPE of BRUSH))
           (SETQ COLOR (fetch BRUSHCOLOR of BRUSH)))
          (T ;; If FUNCTIONAL BRUSH then BIG trouble!
             (printout T T "[In \DRAWCURVE.PSC: Functional BRUSH not supported.]
                   [Using (ROUND 1) BRUSH]" T)
             (SETQ WIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA))
             (SETQ SHAPE 'ROUND]
       (COND
          ((NOT (ZEROP WIDTH))
           (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
           (COND
              ((FLOATP COLOR)
               (POSTSCRIPT.PUTCOMMAND STREAM COLOR " setgray ")
               ;; COLOR is specified in POSTSCRIPT setgray notation.

               ))
           (COND
              ((LISTP DASHING)
               (POSTSCRIPT.OUTSTR STREAM " [")
               (for D in DASHING do (POSTSCRIPT.PUTCOMMAND STREAM (TIMES D WIDTH)
                                           " ")
                                    ;; Since Interlisp DASHING are in terms of BRUSH units, we must multiply by the brush size.

                                    )
               (POSTSCRIPT.PUTCOMMAND STREAM "] 0 setdash" :EOL)))
           (SETQ PSPLINE (PARAMETRICSPLINE KNOTS CLOSED NIL))
           (SETQ N (pop PSPLINE))
           (SETQ XA (pop PSPLINE))
           (SETQ YA (pop PSPLINE))
           (SETQ DXA (pop PSPLINE))
           (SETQ DYA (pop PSPLINE))
           (POSTSCRIPT.PUTCOMMAND STREAM (SELECTQ SHAPE
                                             (ROUND " 1 setlinecap 1 setlinejoin ")
                                             (SQUARE " 2 setlinecap 0 setlinejoin ")
                                             " 0 setlinecap 0 setlinejoin ")
                  WIDTH " setlinewidth " (SETQ PREVX (ELT XA 1))
                  " "
                  (SETQ PREVY (ELT YA 1))
                  " M" :EOL)
           (SETQ PREV-DX3 (FQUOTIENT (ELT DXA 1)
                                     3.0))
           (SETQ PREV-DY3 (FQUOTIENT (ELT DYA 1)
                                     3.0))
           (for C from 2 to N do (POSTSCRIPT.PUTCOMMAND STREAM (FPLUS PREVX PREV-DX3)
                                        " "
                                        (FPLUS PREVY PREV-DY3)
                                        " "
                                        (FDIFFERENCE (SETQ PREVX (ELT XA C))
                                               (SETQ PREV-DX3 (FQUOTIENT (ELT DXA C)
                                                                         3.0)))
                                        " "
                                        (FDIFFERENCE (SETQ PREVY (ELT YA C))
                                               (SETQ PREV-DY3 (FQUOTIENT (ELT DYA C)
                                                                         3.0)))
                                        " " PREVX " " PREVY " curveto" :EOL))
           (POSTSCRIPT.PUTCOMMAND STREAM "stroke" :EOL "grestore" :EOL)))
       (\MOVETO.PSC STREAM PREVX PREVY))
  NIL])
```

(\DRAWELLIPSE.PSC

```
[LAMBDA (STREAM CENTERX CENTERY MINORRADIUS MAJORRADIUS ORIENTATION BRUSH DASHING)
                                                               ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        WIDTH COLOR)
       [COND
          ((NUMBERP BRUSH)
           (SETQ WIDTH BRUSH))
          ((LISTP BRUSH)
           (COND
              ((NEQ (fetch BRUSHSHAPE of BRUSH)
                    'ROUND)
```

```
                (printout T T "[In \DRAWELLIPSE.PSC: Non-ROUND BRUSH not supported.]
                      [Using ROUND BRUSH]" T)))
            (SETQ WIDTH (fetch BRUSHSIZE of BRUSH))
            (SETQ COLOR (fetch BRUSHCOLOR of BRUSH)))
           (T                                                    ; If FUNCTIONAL BRUSH, big trouble!
              (printout T T "[In \DRAWELLIPSE.PSC: Functional BRUSH not supported.]
                    [Using (ROUND 1) BRUSH]" T)
              (SETQ WIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA]
        (COND
           ((NOT (ZEROP WIDTH))
            (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
            (COND
               ((FLOATP COLOR)
                (POSTSCRIPT.PUTCOMMAND STREAM COLOR " setgray ")
                                                                 ; COLOR is specified in POSTSCRIPT setgray notation.

               ))
            (COND
               ((LISTP DASHING)
                (POSTSCRIPT.OUTSTR STREAM " [")
                (for D in DASHING do (POSTSCRIPT.PUTCOMMAND STREAM (TIMES D WIDTH)
                                            " ")
                                     ;; Since Interlisp DASHING are in terms of BRUSH units, we must multiply by the brush size.

                                     )
                (POSTSCRIPT.PUTCOMMAND STREAM "] 0 setdash" :EOL)))
            (POSTSCRIPT.PUTCOMMAND STREAM WIDTH " setlinewidth 1 setlinecap 1 setlinejoin " CENTERX " "
                  CENTERY " " MAJORRADIUS " " MINORRADIUS " " ORIENTATION " 0 360 ellipse stroke" :EOL
                  "grestore" :EOL)))
        (\MOVETO.PSC STREAM CENTERX CENTERY])
```

(\DRAWLINE.PSC

```
[LAMBDA (STREAM X1 Y1 X2 Y2 WIDTH OPERATION COLOR DASHING)      ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
```

;; DRAWLINE method for postscript streams.

```
(LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
     [COND
        ((NOT (NUMBERP WIDTH))
```

;; The WIDTH = NIL should have been handled before here, but just in case!

```
         (SETQ WIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA]
     [COND
        ((NOT (ZEROP WIDTH))
         (COND
            ((LESSP X2 X1)
```

;; For Syntelligence, make all lines move from left to right, to defeat a bug in SPARCPrinter PS decoder.

```
             (\DRAWLINE.PSC STREAM X2 Y2 X1 Y1 WIDTH OPERATION COLOR DASHING))
            ((NOT (OR (FLOATP COLOR)
                      (LISTP DASHING)))                          ; Simple case, no dash or gray
             (POSTSCRIPT.PUTCOMMAND STREAM X2 " " Y2 " " X1 " " Y1 " " WIDTH " L" :EOL))
            (T                                                   ; COLOR is interpreted as gray factor
               (POSTSCRIPT.PUTCOMMAND STREAM X2 " " Y2 " " X1 " " Y1 " " WIDTH " " (OR (FLOATP COLOR)
                                                                                      "0")
                     " [")
               (for D in (LISTP DASHING) do                      ;; Interlisp DASHING is in terms of BRUSH units, so multiply by the brush size.
                                            (POSTSCRIPT.PUTCOMMAND STREAM (TIMES D WIDTH)
                                                   " "))
               (POSTSCRIPT.PUTCOMMAND STREAM "] L1" :EOL]
     (replace (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA with X2)
     (freplace (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA with Y2)
     (freplace (\POSTSCRIPTDATA POSTSCRIPTMOVEFLG) of IMAGEDATA with NIL])
```

(\DRAWPOINT.PSC

```
[LAMBDA (STREAM X Y BRUSH OPERATION)                            ; Edited 30-Mar-90 17:53 by Matt Heffron
```

;; draw a point on the stream

```
(if (BITMAPP BRUSH)
    then (LET ((WIDTH (fetch BITMAPWIDTH of BRUSH))
               (HEIGHT (fetch BITMAPHEIGHT of BRUSH)))
              (BITBLT BRUSH 0 0 STREAM (- X (IQUOTIENT WIDTH 2))
                     (- Y (IQUOTIENT HEIGHT 2))
                     WIDTH HEIGHT OPERATION))
  else (\DRAWLINE.PSC STREAM X Y X Y BRUSH OPERATION])
```

(\DRAWPOLYGON.PSC

```
[LAMBDA (STREAM POINTS CLOSED BRUSH DASHING)                    ; Edited 20-Nov-92 15:17 by sybalsky:mv:envos
  (LET ((LASTPOINT (CAR (LAST POINTS)))
        (IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        WIDTH SHAPE COLOR)
       [COND
          ((NUMBERP BRUSH)
           (SETQ WIDTH BRUSH)
```

```
            (SETQ SHAPE 'ROUND))
           ((LISTP BRUSH)
            (SETQ WIDTH (fetch BRUSHSIZE of BRUSH))
            (SETQ SHAPE (fetch BRUSHSHAPE of BRUSH))
            (SETQ COLOR (fetch BRUSHCOLOR of BRUSH)))
           (T
              ;; If FUNCTIONAL BRUSH then BIG trouble!
              (printout T T "[In \DRAWPOLYGON.PSC: Functional BRUSH not supported.]
                     [Using (ROUND 1) BRUSH]" T)
              (SETQ WIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA))
              (SETQ SHAPE 'ROUND]
        (COND
           ((NOT (ZEROP WIDTH))
            (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
            (COND
               ((FLOATP COLOR)
                (POSTSCRIPT.PUTCOMMAND STREAM COLOR " setgray ")
                ;; COLOR is specified in POSTSCRIPT setgray notation.

                ))
            (COND
               ((LISTP DASHING)
                (POSTSCRIPT.OUTSTR STREAM " [")
                (for D in DASHING do (POSTSCRIPT.PUTCOMMAND STREAM (TIMES D WIDTH)
                                                            " ")
                                     ;; Since Interlisp DASHING are in terms of BRUSH units, we must multiply by the brush size.

                                     )
                (POSTSCRIPT.PUTCOMMAND STREAM "] 0 setdash" :EOL)))
            (POSTSCRIPT.PUTCOMMAND STREAM (SELECTQ SHAPE
                                                   (ROUND " 1 setlinecap 1 setlinejoin ")
                                                   (SQUARE " 2 setlinecap 0 setlinejoin ")
                                                   " 0 setlinecap 0 setlinejoin ")
                  WIDTH " setlinewidth " (fetch (POSITION XCOORD) of (CAR POINTS))
                  " "
                  (fetch (POSITION YCOORD) of (CAR POINTS))
                  " M" :EOL)
            (for P in (CDR POINTS) do (POSTSCRIPT.PUTCOMMAND STREAM (fetch (POSITION XCOORD) of P)
                                                             " "
                                                             (fetch (POSITION YCOORD) of P)
                                                             " lineto" :EOL))
            (COND
               (CLOSED (POSTSCRIPT.PUTCOMMAND STREAM " closepath")))
            (POSTSCRIPT.PUTCOMMAND STREAM " stroke" :EOL "grestore" :EOL)))
        (\MOVETO.PSC STREAM (fetch (POSITION XCOORD) of LASTPOINT)
               (fetch (POSITION YCOORD) of LASTPOINT])
```

(\DSPBOTTOMMARGIN.PSC

```
[LAMBDA (STREAM YPOSITION)                                  ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (PROG1 (fetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of (fetch (STREAM IMAGEDATA) of STREAM))
      (COND
         (YPOSITION (replace (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of (fetch (STREAM IMAGEDATA) of STREAM)
                       with YPOSITION)))))])
```

(\DSPCLIPPINGREGION.PSC

```
[LAMBDA (STREAM REGION)                                     ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OLDCLIP (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA)))
        (COND
           ([AND REGION (NOT (AND (EQP (fetch (REGION LEFT) of OLDCLIP)
                                       (fetch (REGION LEFT) of REGION))
                                  (EQP (fetch (REGION BOTTOM) of OLDCLIP)
                                       (fetch (REGION BOTTOM) of REGION))
                                  (EQP (fetch (REGION WIDTH) of OLDCLIP)
                                       (fetch (REGION WIDTH) of REGION))
                                  (EQP (fetch (REGION HEIGHT) of OLDCLIP)
                                       (fetch (REGION HEIGHT) of REGION]
            (POSTSCRIPT.SHOWACCUM STREAM)
            (replace (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA with REGION)
            (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with T)
            (\FIXLINELENGTH.PSC STREAM IMAGEDATA)))
         OLDCLIP])
```

(\DSPCOLOR.PSC

```
[LAMBDA (STREAM COLOR)                                      ; Edited 14-Jan-93 17:14 by jds
  ;; Postscript "color" setter -- really sets gray shade for now.  0.0 = black, 1.0 = white.
  (POSTSCRIPT.SHOWACCUM STREAM)
  (PROG1 (FETCH (\POSTSCRIPTDATA POSTSCRIPTCOLOR) OF (FETCH (STREAM IMAGEDATA) OF STREAM))
      (COND
         ((AND (NUMBERP COLOR)
               (<= 0 COLOR 1))
          (REPLACE (\POSTSCRIPTDATA POSTSCRIPTCOLOR) OF (FETCH (STREAM IMAGEDATA) OF STREAM) WITH COLOR)
```

```
            (POSTSCRIPT.PUTCOMMAND STREAM :EOL COLOR " setgray "))
           (COLOR (\ILLEGAL.ARG COLOR))))])
```

(\DSPFONT.PSC

```
[LAMBDA (STREAM FONT)                                   ; Edited 26-May-93 01:06 by sybalsky:mv:envos
                                                        ; Edited 11-May-93 02:11 by jds
                                                        ; Edited 19-Jan-93 17:17 by jds
    ;; Change fonts on the PostScript stream STREAM to be FONT.
    ;; Doesn't actually write the font-change command to the stream (it saves doing that until the font is actually needed, so that multiple font changes
    ;; don't yield larger PS files).
    (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (OLDFONT (fetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of IMAGEDATA))
           NEWFONT FONTID)
          (COND
             ((AND FONT (SETQ NEWFONT (OR (\COERCEFONTDESC FONT STREAM)
                                          (FONTCOPY OLDFONT FONT)))
                   (type? FONTDESCRIPTOR NEWFONT)
                   (NEQ NEWFONT OLDFONT))
              ;; OK, it's a good font.
              (POSTSCRIPT.SHOWACCUM STREAM)             ; Write out any accumulated characters.
              ;; Change the font in the Lisp stream:
              (replace (\POSTSCRIPTDATA POSTSCRIPTFONT) of IMAGEDATA with NEWFONT)
              ;; and now update all font-dependent fields in the imagedata, EXCEPT POSTSCRIPTSPACEWIDTH and
              ;; POSTSCRIPTNATURALSPACEWIDTH.  These latter 2 must stay as-is up thru the actual writing of characters by SHOWACCUM,
              ;; so
              (\POSTSCRIPT.CHANGECHARSET IMAGEDATA 0)
              (\DSPLINEFEED.PSC STREAM (IMINUS (fetch (FONTDESCRIPTOR \SFHeight) of NEWFONT)))
              [replace (\POSTSCRIPTDATA POSTSCRIPTSPACEWIDTH) of IMAGEDATA
                 with (FIXR (TIMES (fetch (\POSTSCRIPTDATA POSTSCRIPTSPACEFACTOR) of IMAGEDATA)
                                   (replace (\POSTSCRIPTDATA POSTSCRIPTNATURALSPACEWIDTH) of IMAGEDATA
                                      with (\FGETWIDTH (fetch (\POSTSCRIPTDATA POSTSCRIPTWIDTHS) of IMAGEDATA)
                                                       (CHARCODE SPACE]
              (\FIXLINELENGTH.PSC STREAM IMAGEDATA)
              [SETQ FONTID (fetch (PSCFONT IL-FONTID) of (LISTGET (fetch (FONTDESCRIPTOR OTHERDEVICEFONTPROPS)
                                                                     of NEWFONT)
                                                                  'PSCFONT]
              (COND
                 ((MEMB (fetch (FONTID FONTIDNAME) of FONTID)
                        *POSTSCRIPT-UNACCENTED-FONTS*)
                  (FREPLACE (\POSTSCRIPTDATA POSTSCRIPTACCENTED) OF IMAGEDATA WITH NIL))
                 (T (freplace (\POSTSCRIPTDATA POSTSCRIPTACCENTED) of IMAGEDATA with T)))
              ;; Remember to actually write a change command
              (replace (\POSTSCRIPTDATA POSTSCRIPTFONTCHANGEDFLG) of IMAGEDATA with T)))
           OLDFONT])
```

(\DSPLEFTMARGIN.PSC

```
[LAMBDA (STREAM XPOSITION)                              ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
       (PROG1 (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
           (COND
              (XPOSITION (replace (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA with XPOSITION)
                     (\FIXLINELENGTH.PSC STREAM IMAGEDATA))))])
```

(\DSPLINEFEED.PSC

```
[LAMBDA (STREAM LINELEADING)                            ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (PROG1 (fetch (\POSTSCRIPTDATA POSTSCRIPTLINESPACING) of (fetch (STREAM IMAGEDATA) of STREAM))
      (COND
         (LINELEADING (replace (\POSTSCRIPTDATA POSTSCRIPTLINESPACING) of (fetch (STREAM IMAGEDATA) of STREAM)
                         with LINELEADING))))])
```

(\DSPPUSHSTATE.PSC

```
[LAMBDA (STREAM)                                        ; Edited 20-Nov-92 15:12 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
       (push (fetch (\POSTSCRIPTDATA POSTSCRIPTXFORMSTACK) of IMAGEDATA)
             (create POSTSCRIPTXFORM
                     PSXCLIP _ (COPY (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA))
                     PSXPAGE _ (COPY (fetch (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA))
                     PSXLEFT _ (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
                     PSXRIGHT _ (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA)
                     PSXTOP _ (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA)
                     PSXBOTTOM _ (fetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA)
                     PSXTRANX _ (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA)
                     PSXTRANY _ (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA)
                     PSXLAND _ (fetch (\POSTSCRIPTDATA POSTSCRIPTLANDSCAPE) of IMAGEDATA)
                     PSXXFORMPEND _ (fetch (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA])
```

(\DSPPOPSTATE.PSC

```
[LAMBDA (STREAM)                                                    ; Edited 20-Nov-92 15:15 by sybalsky:mv:envos
  (LET* [(IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (XFORM (pop (fetch (\POSTSCRIPTDATA POSTSCRIPTXFORMSTACK) of IMAGEDATA]
        (replace (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXCLIP)
                                                                                  of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXPAGE)
                                                                              of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXBOTTOM)
                                                                                of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXTOP)
                                                                             of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXLEFT)
                                                                              of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXRIGHT)
                                                                               of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTLANDSCAPE) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXLAND)
                                                                             of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXXFORMPEND)
                                                                                of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXTRANX)
                                                                           of XFORM))
        (replace (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA with (fetch (POSTSCRIPTXFORM PSXTRANY)
                                                                           of XFORM])
```

(\DSPRESET.PSC

```
[LAMBDA (STREAM)                                                    ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
       (replace (STREAM CHARPOSITION) of STREAM with 0)
       (\MOVETO.PSC STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
              (DIFFERENCE (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA)
                     (FONTPROP (fetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of IMAGEDATA)
                            'ASCENT])
```

(\DSPRIGHTMARGIN.PSC

```
[LAMBDA (STREAM XPOSITION)                                          ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
       (PROG1 (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA)
           (COND
              (XPOSITION (replace (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA with XPOSITION)
                     (\FIXLINELENGTH.PSC STREAM IMAGEDATA))))])
```

(\DSPROTATE.PSC

```
[LAMBDA (STREAM ROTATION)                                           ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
```

;; rotate the postscript stream by ROTATION

;; we only know 90 degrees of rotation for now (0 means portrait, anything else is landscape).

```
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OROT (fetch (\POSTSCRIPTDATA POSTSCRIPTROTATION) of IMAGEDATA))
         LAND C0 P0 C P ML MB MR MT)
        (COND
           ((AND ROTATION (NEQ ROTATION (fetch (\POSTSCRIPTDATA POSTSCRIPTROTATION) of IMAGEDATA)))
            (POSTSCRIPT.SHOWACCUM STREAM)
            (replace (\POSTSCRIPTDATA POSTSCRIPTROTATION) of IMAGEDATA with ROTATION)
            (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with T)
            (\DSPRESET.PSC STREAM)))
        OROT])
```

(\DSPSCALE.PSC

```
[LAMBDA (STREAM SCALE)                                              ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OSCALE (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA))
         NSCALE)
        (COND
           ((AND NIL
```

;; Changing SCALE is not implemented. According to IRM.

```
                 (NUMBERP SCALE)
                 (CL:PLUSP SCALE))
            (SETQ NSCALE (QUOTIENT SCALE OSCALE))
```

;; NSCALE is the adjustment for the fact that the scale operator takes RELATIVE scale changes.

```
            (POSTSCRIPT.PUTCOMMAND STREAM " " NSCALE " " NSCALE " scale" :EOL)
            (replace (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA with SCALE)))
        OSCALE])
```

(\DSPSCALE2.PSC

```
[LAMBDA (STREAM X-SCALE Y-SCALE)                                    ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
```

;; SETS X AND Y SCALE

```
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OSCALE (fetch (\POSTSCRIPTDATA POSTSCRIPTSCALE) of IMAGEDATA))
         NSCALE)
```

```
          (COND
             ((AND X-SCALE (NUMBERP X-SCALE)
                   (CL:PLUSP X-SCALE))
              (POSTSCRIPT.SHOWACCUM STREAM)
              (\UPDATE.PSC STREAM IMAGEDATA)
              ;; NSCALE is the adjustment for the fact that the scale operator takes RELATIVE scale changes.
              (POSTSCRIPT.PUTCOMMAND STREAM " " X-SCALE " " Y-SCALE " scale" :EOL)))
          T])
```

(\DSPSPACEFACTOR.PSC

```
[LAMBDA (STREAM FACTOR)                                    ; Edited 26-May-93 01:18 by sybalsky:mv:envos
  (DECLARE (LOCALVARS . T))
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (OLDFACTOR (fetch (\POSTSCRIPTDATA POSTSCRIPTSPACEFACTOR) of IMAGEDATA)))
        [COND
           ((AND (NUMBERP FACTOR)
                 (NOT (EQUAL FACTOR OLDFACTOR)))
            (POSTSCRIPT.SHOWACCUM STREAM)
            (replace (\POSTSCRIPTDATA POSTSCRIPTSPACEFACTOR) of IMAGEDATA with FACTOR)
            (replace (\POSTSCRIPTDATA POSTSCRIPTSPACEWIDTH) of IMAGEDATA
               with (FIXR (TIMES FACTOR (ffetch (\POSTSCRIPTDATA POSTSCRIPTNATURALSPACEWIDTH) of IMAGEDATA]
        OLDFACTOR])
```

(\DSPTOPMARGIN.PSC

```
[LAMBDA (STREAM YPOSITION)                                 ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (PROG1 (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of (fetch (STREAM IMAGEDATA) of STREAM))
      (COND
         (YPOSITION (replace (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of (fetch (STREAM IMAGEDATA) of STREAM)
                       with YPOSITION)))])
```

(\DSPTRANSLATE.PSC

```
[LAMBDA (STREAM TX TY)                                     ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
         (MDX (DIFFERENCE (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA)
                     TX))
         (MDY (DIFFERENCE (fetch (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA)
                     TY)))
        (COND
           ((NOT (AND (ZEROP MDX)
                      (ZEROP MDY)))
            (POSTSCRIPT.SHOWACCUM STREAM)
            (for REG in (LIST (fetch (\POSTSCRIPTDATA POSTSCRIPTCLIPPINGREGION) of IMAGEDATA)
                              (fetch (\POSTSCRIPTDATA POSTSCRIPTPAGEREGION) of IMAGEDATA))
               do (CL:INCF (fetch (REGION LEFT) of REG)
                           MDX)
                  (CL:INCF (fetch (REGION BOTTOM) of REG)
                           MDY))
            (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                     MDX)
            (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA)
                     MDY)
            (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA)
                     MDX)
            (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA)
                     MDX)
            (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTBOTTOMMARGIN) of IMAGEDATA)
                     MDY)
            (CL:INCF (fetch (\POSTSCRIPTDATA POSTSCRIPTTOPMARGIN) of IMAGEDATA)
                     MDY)
            (replace (\POSTSCRIPTDATA POSTSCRIPTTRANSX) of IMAGEDATA with TX)
            (replace (\POSTSCRIPTDATA POSTSCRIPTTRANSY) of IMAGEDATA with TY)
            (replace (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA with T])
```

(\DSPXPOSITION.PSC

```
[LAMBDA (STREAM XPOSITION)                                 ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        OLDX)
       (PROG1 (SETQ OLDX (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA))
           [COND
              ((AND XPOSITION (NOT (EQUAL XPOSITION OLDX)))
               (\MOVETO.PSC STREAM XPOSITION (fetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA])])
```

(\DSPYPOSITION.PSC

```
[LAMBDA (STREAM YPOSITION)                                 ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
  (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
        OLDY)
       (PROG1 (SETQ OLDY (fetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA))
           (COND
              ((AND YPOSITION (NOT (EQUAL YPOSITION OLDY)))
               (\MOVETO.PSC STREAM (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                      YPOSITION)))])
```

(\FILLCIRCLE.PSC

```
[LAMBDA (STREAM CENTERX CENTERY RADIUS TEXTURE)                    ; Edited 30-Mar-90 17:59 by Matt Heffron
  (LET (TEXTUREBM TEXTUREWIDTH)
       (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
       (if (FIXP TEXTURE)
           then (if (ZEROP TEXTURE)
                    then (SETQ TEXTURE 1.0)                        ; The setgray version of white
                  elseif (OR (EQL TEXTURE 65535)
                             (EQL TEXTURE -1))
                    then (SETQ TEXTURE 0.0)                        ; The setgray version of black
                    ))
       (if (FLOATP TEXTURE)
           then ;; If TEXTURE is a FLOATP, then it is specified in PostScript setgray notation.
                (POSTSCRIPT.PUTCOMMAND STREAM TEXTURE " setgray ")
         elseif (OR (TEXTUREP TEXTURE)
                    (NULL TEXTURE))
           then (SETQ TEXTUREBM (BITMAPCREATE 16 16 1))
                (SETQ TEXTUREWIDTH 16)
                (BLTSHADE TEXTURE TEXTUREBM)
         elseif (BITMAPP TEXTURE)
           then (SETQ TEXTUREWIDTH (MIN (fetch BITMAPWIDTH of TEXTUREBM)
                                        (fetch BITMAPHEIGHT of TEXTUREBM)))
                (SETQ TEXTUREBM (BITMAPCREATE TEXTUREWIDTH TEXTUREWIDTH 1))
                (BITBLT TEXTURE 0 0 TEXTUREBM 0 0 TEXTUREWIDTH TEXTUREWIDTH 'INPUT 'REPLACE))
       (POSTSCRIPT.PUTCOMMAND STREAM " " CENTERX " " CENTERY " " RADIUS " 0 360 arc" :EOL)
       (if TEXTUREBM
           then (POSTSCRIPT.PUTCOMMAND STREAM "100 100 scale ")
                (POSTSCRIPT.PUTBITMAPBYTES STREAM TEXTUREBM T)
                (POSTSCRIPT.PUTCOMMAND STREAM TEXTUREWIDTH " " (LSH (fetch BITMAPRASTERWIDTH of TEXTUREBM)
                                                                    1)
                       " 0 "
                       (TIMES 72 (QUOTIENT (DSPSCALE NIL STREAM)
                                           100.0))
                       " findresolution " TEXTUREWIDTH " div div ceiling " POSTSCRIPT.TEXTURE.SCALE " mul
                       setpattern eofill" :EOL "grestore" :EOL)
         else (POSTSCRIPT.PUTCOMMAND STREAM " eofill" :EOL "grestore" :EOL))
       (\MOVETO.PSC STREAM CENTERX CENTERY])
```

(\FILLPOLYGON.PSC

```
[LAMBDA (STREAM KNOTS TEXTURE OPERATION WINDNUMBER)                ; Edited 20-Nov-92 15:17 by sybalsky:mv:envos
  (DECLARE (SPECVARS FILL.WRULE))
  ;; OPERATION is ignored here
  (LET ((LASTPOINT (CAR (LAST KNOTS)))
        TEXTUREBM TEXTUREWIDTH)
       (POSTSCRIPT.PUTCOMMAND STREAM :EOL "gsave newpath ")
       (if (NOT (OR (ZEROP WINDNUMBER)
                    (EQL WINDNUMBER 1)))
           then (SETQ WINDNUMBER FILL.WRULE))
       (if (FIXP TEXTURE)
           then (if (ZEROP TEXTURE)
                    then (SETQ TEXTURE 1.0)                        ; The setgray version of white
                  elseif (OR (EQL TEXTURE 65535)
                             (EQL TEXTURE -1))
                    then (SETQ TEXTURE 0.0)                        ; The setgray version of black
                    ))
       (if (FLOATP TEXTURE)
           then ;; If TEXTURE is a FLOATP, then it is specified in PostScript setgray notation.
                (POSTSCRIPT.PUTCOMMAND STREAM TEXTURE " setgray ")
         elseif (OR (TEXTUREP TEXTURE)
                    (NULL TEXTURE))
           then (SETQ TEXTUREBM (BITMAPCREATE 16 16 1))
                (SETQ TEXTUREWIDTH 16)
                (BLTSHADE TEXTURE TEXTUREBM)
         elseif (BITMAPP TEXTURE)
           then (SETQ TEXTUREWIDTH (MIN (fetch BITMAPWIDTH of TEXTUREBM)
                                        (fetch BITMAPHEIGHT of TEXTUREBM)))
                (SETQ TEXTUREBM (BITMAPCREATE TEXTUREWIDTH TEXTUREWIDTH 1))
                (BITBLT TEXTURE 0 0 TEXTUREBM 0 0 TEXTUREWIDTH TEXTUREWIDTH 'INPUT 'REPLACE))
       (POSTSCRIPT.PUTCOMMAND STREAM (fetch (POSITION XCOORD) of (CAR KNOTS))
              " "
              (fetch (POSITION YCOORD) of (CAR KNOTS))
              " M" :EOL)
       (for K in (CDR KNOTS) do (POSTSCRIPT.PUTCOMMAND STREAM (fetch (POSITION XCOORD) of K)
                                       " "
                                       (fetch (POSITION YCOORD) of K)
                                       " lineto" :EOL))
       (POSTSCRIPT.PUTCOMMAND STREAM " closepath" :EOL)
       (if TEXTUREBM
           then (POSTSCRIPT.PUTCOMMAND STREAM "100 100 scale ")
                (POSTSCRIPT.PUTBITMAPBYTES STREAM TEXTUREBM T)
                (POSTSCRIPT.PUTCOMMAND STREAM TEXTUREWIDTH " " (LSH (fetch BITMAPRASTERWIDTH of TEXTUREBM)
```

```
                                                                    1)
                        " 0 "
                        (TIMES 72 (QUOTIENT (DSPSCALE NIL STREAM)
                                            100.0))
                        " findresolution " TEXTUREWIDTH " div div ceiling " POSTSCRIPT.TEXTURE.SCALE " mul
                        setpattern"))
        (POSTSCRIPT.PUTCOMMAND STREAM (if (ZEROP WINDNUMBER)
                                          then " fill"
                                        else " eofill")
              :EOL "grestore" :EOL)
        (\MOVETO.PSC STREAM (fetch (POSITION XCOORD) of LASTPOINT)
               (fetch (POSITION YCOORD) of LASTPOINT])
```

(\FIXLINELENGTH.PSC

```
  [LAMBDA (STREAM IMAGEDATA)                                        ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
```

;; Called by margin, font or rotation change to update the LINELENGTH field in the stream.

```
    (LET [(TMP (MIN MAX.SMALLP (FIX (QUOTIENT (DIFFERENCE (fetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN)
                                                                of IMAGEDATA)
                                                          (ffetch (\POSTSCRIPTDATA POSTSCRIPTLEFTMARGIN) of IMAGEDATA))
                                              (fetch FONTAVGCHARWIDTH of (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONT)
                                                                                 of IMAGEDATA]
          (replace (STREAM LINELENGTH) of STREAM with (COND
                                                        ((GREATERP TMP 1)
                                                         TMP)
                                                        (T 10])
```

(\MOVETO.PSC

```
  [LAMBDA (STREAM X Y)                                              ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
    (LET ((IMAGEDATA (ffetch (STREAM IMAGEDATA) of STREAM)))
         (COND
            ([NOT (AND (EQP X (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA))
                       (EQP Y (ffetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA]
             (POSTSCRIPT.SHOWACCUM STREAM)
             (freplace (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA with X)
             (freplace (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA with Y)
             (freplace (\POSTSCRIPTDATA POSTSCRIPTMOVEFLG) of IMAGEDATA with T])
```

(\NEWPAGE.PSC

```
  [LAMBDA (STREAM)                                                  ; Edited  5-Apr-89 17:31 by TAL
    (POSTSCRIPT.ENDPAGE STREAM)
    (POSTSCRIPT.STARTPAGE STREAM])

)
```

;; Character-output, plus special-cases:

```
(DEFINEQ
```

(\POSTSCRIPT.CHANGECHARSET

```
  [LAMBDA (PSDATA CHARSET)                                          ; Edited 29-Apr-93 13:51 by rmk:
```

;; Called when the character set information cached in a display stream doesn't correspond to CHARSET

```
    (PROG* ((FONT (ffetch POSTSCRIPTFONT of PSDATA))
            (CSINFO (\GETCHARSETINFO CHARSET FONT)))
```

;; since the call to \getcharsetinfo has NOSLUG? = NIL, we know that we will get a reasonable character set back

```
           (UNINTERRUPTABLY
               (freplace POSTSCRIPTWIDTHS of PSDATA with (ffetch (CHARSETINFO WIDTHS) of CSINFO))
               (freplace POSTSCRIPTNSCHARSET of PSDATA with CHARSET))])
```

(\POSTSCRIPT.OUTCHARFN

```
  [LAMBDA (STREAM CHAR)                                             ; Edited 23-May-93 12:00 by rmk:
                                                                    ; Edited  4-May-93 02:20 by jds
                                                                    ; Edited  3-Feb-93 00:45 by jds
```

;;; Output a character to be printed.

;;; Change font if necessary, do newline if at right margin, check for special chars and do appropriate thing, quote char and/or start postscript string if ;;; necessary.

;;; This is called a lot, so the code is unrolled for efficiency.

```
    (DECLARE (GLOBALVARS \POSTSCRIPT.CHARTYPE)
             (LOCALVARS . T))
    (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (XPOS (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA))
           (FONT (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of IMAGEDATA))
           CHARWID NEWXPOS MAPPING)
          (CL:UNLESS (EQ (\CHARSET CHAR)
                         (ffetch POSTSCRIPTNSCHARSET of IMAGEDATA))
```

;; Switch character set so that we get the right char width.

```
        (\POSTSCRIPT.CHANGECHARSET IMAGEDATA (\CHARSET CHAR)))
    [SETQ CHARWID (SELCHARQ CHAR
                        (SPACE (ffetch (\POSTSCRIPTDATA POSTSCRIPTSPACEWIDTH) of IMAGEDATA))
                        (\FGETWIDTH (ffetch (\POSTSCRIPTDATA POSTSCRIPTWIDTHS) of IMAGEDATA)
                               (\CHAR8CODE CHAR]
;; POSTSCRIPTACCENTED true if font has accented rendering characters in it; otherwise, a c-set 0 special font (SYMBOL,
;; ZAPFDINGBATS...)
    [COND
       [[OR (NOT (ffetch (\POSTSCRIPTDATA POSTSCRIPTACCENTED) of IMAGEDATA))
            (AND (ILEQ CHAR 254)
                 (NOT (CL:AREF \POSTSCRIPT.CHARTYPE CHAR]
        ;; OR is NIL if char is special in any way:  Either font isn't supposed to be treated as an NS font (e.g. ZapfDingbats, which uses all the
        ;; legal char positions for its own), or char itself is in cset 0 and ordinary
        [COND
           ((IGREATERP (SETQ NEWXPOS (IPLUS XPOS CHARWID))
                       (ffetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA))
            (\TERPRI.PSC STREAM)
            (SETQ NEWXPOS (IPLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                                 CHARWID]
        (CL:UNLESS (ffetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA)
            (\UPDATE.PSC STREAM IMAGEDATA)
            (BOUT STREAM (CHARCODE %())
            (freplace (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA with T))
        (COND
           [(ILESSP CHAR (CHARCODE " "))
            (BOUT STREAM (CHARCODE \))
            [BOUT STREAM (IPLUS (CHARCODE 0)
                                (LOGAND 3 (LRSH CHAR 6]
            [BOUT STREAM (IPLUS (CHARCODE 0)
                                (LOGAND 7 (LRSH CHAR 3]
            (BOUT STREAM (IPLUS (CHARCODE 0)
                                (LOGAND 7 CHAR]
           [(IGEQ CHAR 127)
            (BOUT STREAM (CHARCODE \))
            [BOUT STREAM (IPLUS (CHARCODE 0)
                                (LOGAND 3 (LRSH CHAR 6]
            [BOUT STREAM (IPLUS (CHARCODE 0)
                                (LOGAND 7 (LRSH CHAR 3]
            (BOUT STREAM (IPLUS (CHARCODE 0)
                                (LOGAND 7 CHAR]
           (T (SELCHARQ CHAR
                   ((%( %) \)
                        (BOUT STREAM (CHARCODE \))
                        (BOUT STREAM CHAR))
                   (BOUT STREAM CHAR]
       [(SETQ MAPPING (GETHASH CHAR *POSTSCRIPT-NS-HASH*)) ; Special character that's taken care of by the NS mapping.
        [COND
           ((IGREATERP (SETQ NEWXPOS (IPLUS XPOS CHARWID))
                       (ffetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA))
            (\TERPRI.PSC STREAM)
            (SETQ NEWXPOS (IPLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                                 CHARWID]
        (SELECTQ (CAR MAPPING)
            (NIL ;; just a remap within the lower 256.  But the code in (CDR MAPPING) is in charset 2 to prevent recursion
                 (\POSTSCRIPT.SPECIALOUTCHARFN STREAM (CADR MAPPING)))
            (SYMBOL ;; Its in the SYMBOL font.  Symbol is specified as "2,xxx" rather than "0,xxx" to defeat translations to symbol that go
                    ;; to matching character codes.
                    (\POSTSCRIPT.SPECIALOUTCHARFN STREAM (CADR MAPPING)
                           'SYMBOL))
            (ACCENT                                          ; Special accent mapping we did
                    (\POSTSCRIPT.ACCENTFN STREAM (CADR MAPPING)))
            (ACCENTPAIR                                      ; Given base char & accent, overlap them.
                        (\POSTSCRIPT.ACCENTPAIR STREAM (CADR MAPPING)
                               (CADDR MAPPING)
                               (CADDDR MAPPING)))
            (DINGBAT                                         ; A Zapf dingbat
                     (\POSTSCRIPT.SPECIALOUTCHARFN STREAM (CADR MAPPING)
                            'ZAPFDINGBATS))
            (APPLY* (POSTSCRIPT.SHOWACCUM STREAM)
                    (\UPDATE.PSC STREAM IMAGEDATA)
                    ;; User function can call any stream operations it wants.  At the end, we guarantee that baseline hasn't changed and
                    ;; that xpos is where the widthset it would be.
                    [freplace (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA with (PROG1 (ffetch (\POSTSCRIPTDATA
                                                                                                POSTSCRIPTY)
                                                                                        of IMAGEDATA)
                                                                                      (APPLY* (CADDR MAPPING)
                                                                                             STREAM
                                                                                             (CADR MAPPING)))])
            (FUNCTION                                        ; Done as special PS code.
                      (POSTSCRIPT.SHOWACCUM STREAM)
                      (\UPDATE.PSC STREAM IMAGEDATA)
                      (POSTSCRIPT.OUTSTR STREAM (CADR MAPPING)))
```

```
                (\ILLEGAL.ARG (CAR MAPPING]
            (T                                                    ; Special char
               (SELCHARQ CHAR
                    ((EOL LF)
                         (\TERPRI.PSC STREAM)
                         ;; Set NEWXPOS to current value here and in FF to preserve value after external resetting.
                         (SETQ NEWXPOS (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)))
                    (FF (DSPNEWPAGE STREAM)
                        (SETQ NEWXPOS (fetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)))
                    (TAB (SETQ NEWXPOS (IPLUS XPOS (\POSTSCRIPTTAB IMAGEDATA)))
                         [COND
                            ((IGREATERP NEWXPOS (ffetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA))
                             (\TERPRI.PSC STREAM)
                             (SETQ NEWXPOS (IPLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                                                  (\POSTSCRIPTTAB IMAGEDATA]
                         (\MOVETO.PSC STREAM NEWXPOS (ffetch (\POSTSCRIPTDATA POSTSCRIPTY) of IMAGEDATA)))
                    ("357,140"                                    ; Ballot box, checked
                              [COND
                                 ((IGREATERP (SETQ NEWXPOS (IPLUS XPOS CHARWID))
                                             (ffetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA))
                                  (\TERPRI.PSC STREAM)
                                  (SETQ NEWXPOS (IPLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                                                       CHARWID]
                              (LET ((OLDFONT (\DSPFONT.PSC STREAM)))
                                   (POSTSCRIPT.SHOWACCUM STREAM)
                                   (\DSPFONT.PSC STREAM (LIST 'ZAPFDINGBATS (fetch (FONTDESCRIPTOR FONTSIZE)
                                                                                   of OLDFONT)
                                                              (fetch (FONTDESCRIPTOR FONTFACE) of OLDFONT)))
                                   (\UPDATE.PSC STREAM IMAGEDATA)
                                   (POSTSCRIPT.OUTSTR STREAM " bboxchk ")
                                   (\DSPFONT.PSC STREAM OLDFONT)))
                    (PROGN [COND
                              ((IGREATERP (SETQ NEWXPOS (IPLUS XPOS CHARWID))
                                          (ffetch (\POSTSCRIPTDATA POSTSCRIPTRIGHTMARGIN) of IMAGEDATA))
                               (\TERPRI.PSC STREAM)
                               (SETQ NEWXPOS (IPLUS (ffetch (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA)
                                                    CHARWID]
                           (COND
                              ((IGEQ CHAR 255)
                               ;; If it's 255 or above and we don't know anything about it, print the black box.  Width vector will determine
                               ;; width of box, to maintain consistency.
                               (\POSTSCRIPT.PRINTSLUG STREAM CHAR))
                              (T (SETQ CHAR (\CHAR8CODE CHAR))
                                 (COND
                                    ((NOT (ffetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA))
                                     (\UPDATE.PSC STREAM IMAGEDATA)
                                     (BOUT STREAM (CHARCODE %()
                                     (freplace (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA with T)))
                                 (BOUT STREAM (CHARCODE \))
                                 (SELCHARQ CHAR
                                      ((%( %) \)
                                           (BOUT STREAM CHAR))
                                      (PROGN [BOUT STREAM (IPLUS (CHARCODE 0)
                                                                 (LOGAND 3 (LRSH CHAR 6]
                                             [BOUT STREAM (IPLUS (CHARCODE 0)
                                                                 (LOGAND 7 (LRSH CHAR 3]
                                             (BOUT STREAM (IPLUS (CHARCODE 0)
                                                                 (LOGAND 7 CHAR]
         (freplace (\POSTSCRIPTDATA POSTSCRIPTX) of IMAGEDATA with NEWXPOS)
         CHAR])
```

(**POSTSCRIPT.PRINTSLUG**

```
  [LAMBDA (STREAM CHAR)                                           ; Edited  9-May-93 21:55 by rmk:
                                                                  ; Edited  4-May-93 02:20 by jds
                                                                  ; Edited  3-Feb-93 00:45 by jds
```

;;; Internal function to display a black box for a missing character. Width is taken from widths vector, so that box and charwidth are always consistent.
;;; Caller (\POSTSCRIPT.OUTCHARFN) is responsible for guaranteeing proper caching of widths vector and for measurement and position updating,
;;; although \DRAWLINE.PSC also updates position.

```
    (DECLARE (LOCALVARS . T))
    (LET ((IMAGEDATA (FETCH (STREAM IMAGEDATA) OF STREAM)))
         (\BLTSHADE.PSC BLACKSHADE STREAM (FETCH (\POSTSCRIPTDATA POSTSCRIPTX) OF IMAGEDATA)
                (FETCH (\POSTSCRIPTDATA POSTSCRIPTY) OF IMAGEDATA)
                (\FGETWIDTH (FFETCH (\POSTSCRIPTDATA POSTSCRIPTWIDTHS) OF IMAGEDATA)
                       (\CHAR8CODE CHAR))
                (FETCH (FONTDESCRIPTOR \SFAscent) OF (FETCH (\POSTSCRIPTDATA POSTSCRIPTFONT) OF IMAGEDATA))
                'PAINT)
         (\MOVETO.PSC STREAM (IPLUS (FETCH (\POSTSCRIPTDATA POSTSCRIPTX) OF IMAGEDATA)
                                    (\FGETWIDTH (FFETCH (\POSTSCRIPTDATA POSTSCRIPTWIDTHS) OF IMAGEDATA)
                                           (\CHAR8CODE CHAR)))
                (FETCH (\POSTSCRIPTDATA POSTSCRIPTY) OF IMAGEDATA])
```

(\POSTSCRIPT.SPECIALOUTCHARFN

```
[LAMBDA (STREAM CHAR FAMILY)                                    ; Edited 23-May-93 13:31 by rmk:
                                                                ; Edited  4-May-93 02:20 by jds
                                                                ; Edited  3-Feb-93 00:45 by jds
```

;;; Internal function to output a special character to be printed, changing font if necessary. Width processing is carried out at higher level. If FAMILY is given, switches to that font (SYMBOL, ZAPFDINGBATS) before printing, then switches back.

```
    (DECLARE (LOCALVARS . T))
    (LET* [(IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (OLDFONT (AND FAMILY (\DSPFONT.PSC STREAM]
          (CL:WHEN OLDFONT
              (\DSPFONT.PSC STREAM (LIST FAMILY (fetch (FONTDESCRIPTOR FONTSIZE) of OLDFONT)
                                         (fetch (FONTDESCRIPTOR FONTFACE) of OLDFONT))))
          (CL:UNLESS (ffetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA)
              (\UPDATE.PSC STREAM IMAGEDATA)
              (BOUT STREAM (CHARCODE %())
              (freplace (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA with T))
          [COND
             [(ILESSP CHAR (CHARCODE " "))
              (BOUT STREAM (CHARCODE \))
              [BOUT STREAM (IPLUS (CHARCODE 0)
                                  (LOGAND 3 (LRSH CHAR 6]
              [BOUT STREAM (IPLUS (CHARCODE 0)
                                  (LOGAND 7 (LRSH CHAR 3]
              (BOUT STREAM (IPLUS (CHARCODE 0)
                                  (LOGAND 7 CHAR]
             [(IGEQ CHAR 127)
              (BOUT STREAM (CHARCODE \))
              [BOUT STREAM (IPLUS (CHARCODE 0)
                                  (LOGAND 3 (LRSH CHAR 6]
              [BOUT STREAM (IPLUS (CHARCODE 0)
                                  (LOGAND 7 (LRSH CHAR 3]
              (BOUT STREAM (IPLUS (CHARCODE 0)
                                  (LOGAND 7 CHAR]
             (T (SELCHARQ CHAR
                      ((%( %) \)
                           (BOUT STREAM (CHARCODE \))
                           (BOUT STREAM CHAR))
                      (BOUT STREAM CHAR]
          (CL:WHEN OLDFONT (\DSPFONT.PSC STREAM OLDFONT))
          CHAR])
```

(\UPDATE.PSC

```
[LAMBDA (STREAM IMAGEDATA)                                      ; Edited 20-Nov-92 15:13 by sybalsky:mv:envos
```

;; Make any outstanding font, scale, location updates, prepatory to something that might depend heavily on it. (e.g. before starting to output
;; characters, or making a scale change) ; This code was originally in \POSTSCRIPT.OUTCHAR &c, and
; is here for commonality.

```
    (COND
       ((ffetch (\POSTSCRIPTDATA POSTSCRIPTPENDINGXFORM) of IMAGEDATA)
        (\SETXFORM.PSC STREAM IMAGEDATA)))
    (COND
       ((ffetch (\POSTSCRIPTDATA POSTSCRIPTFONTCHANGEDFLG) of IMAGEDATA)
                                                                ; If font was changed then switch before printing
        (\SWITCHFONTS.PSC STREAM IMAGEDATA)))
    (COND
       ((ffetch (\POSTSCRIPTDATA POSTSCRIPTMOVEFLG) of IMAGEDATA)  ; likewise for position
        (\SETPOS.PSC STREAM IMAGEDATA])
```

(\POSTSCRIPT.ACCENTFN

```
[LAMBDA (STREAM CHAR)                                           ; Edited 28-Apr-93 16:35 by rmk:
                                                                ; Edited  3-Feb-93 01:05 by jds
```

;;; Output an accented character to be printed. .

;;;; Need to inc CHARPOSITION of STREAM

```
    (DECLARE (LOCALVARS . T))
    (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM)))
         (COND
            ((NOT (ffetch (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA))
             (\UPDATE.PSC STREAM IMAGEDATA)
             (BOUT STREAM (CHARCODE %())
             (freplace (\POSTSCRIPTDATA POSTSCRIPTCHARSTOSHOW) of IMAGEDATA with T)))
         (BOUT STREAM (CHARCODE "\"))
         (for CH instring (SUBSTRING (CONCAT "000" (OCTALSTRING CHAR))
                                     -3)
            do (BOUT STREAM CH))
         CHAR])
```

(\POSTSCRIPT.ACCENTPAIR

```
[LAMBDA (STREAM CHAR ACCENTS UNDER-ACCENTS)                     ; Edited 17-Aug-93 17:02 by sybalskY:MV:ENVOS
                                                                ; Edited  3-Feb-93 01:29 by jds
```

;;; Output an accented character to be printed. .

;;;; Prints the character as \xxx, with 3 octal digits, to avoid tripping up on EOLs and other postscript-special characters.

```
    (DECLARE (LOCALVARS . T))
    (LET* ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
           (FONT (ffetch (\POSTSCRIPTDATA POSTSCRIPTFONT) of IMAGEDATA)))
          (POSTSCRIPT.SHOWACCUM STREAM)
          (\UPDATE.PSC STREAM IMAGEDATA)
          (BOUT STREAM (CHARCODE %())
          (BOUT STREAM (CHARCODE "\"))
          (for CH instring (SUBSTRING (CONCAT "000" (OCTALSTRING CHAR))
                                      -3)
             do (BOUT STREAM CH))
          (BOUT STREAM (CHARCODE %)))
          (BOUT STREAM (CHARCODE %())
          (for ACCENT inside ACCENTS do (BOUT STREAM (CHARCODE "\"))
                                        (for CH instring (SUBSTRING (CONCAT "000" (OCTALSTRING ACCENT))
                                                                    -3)
                                           do (BOUT STREAM CH)))
          (POSTSCRIPT.PUTCOMMAND STREAM ") (")
          (for ACCENT inside UNDER-ACCENTS do (BOUT STREAM (CHARCODE "\"))
                                              (for CH instring (SUBSTRING (CONCAT "000" (OCTALSTRING ACCENT))
                                                                          -3)
                                                 do (BOUT STREAM CH)))
          (BOUT STREAM (CHARCODE %)))
          (COND
             (NIL (OR (IEQP ACCENT (CHARCODE "0,313"))
                      (IEQP ACCENT (CHARCODE "0,316")))          ; Cedilla and ogonek are under-accents, so don't raise them for
                                                                 ; capital letters.
                  (POSTSCRIPT.PUTCOMMAND STREAM " 0 "))
             ((ILESSP CHAR (CHARCODE a))                         ; upper case, so adjust offset for accent
              (POSTSCRIPT.PUTCOMMAND STREAM " " (/ (fetch \SFAscent of FONT)
                                                   3.0)
                     " "))
             (T (POSTSCRIPT.PUTCOMMAND STREAM " 0 ")))
          (POSTSCRIPT.PUTCOMMAND STREAM " " (FONTPROP FONT 'SIZE)
                 " ")
          (POSTSCRIPT.PUTCOMMAND STREAM " accentor ")
          CHAR])

)

;; Spacing-character (M-quad, etc.) and ballot-box-check &c special-case functions

(DEFINEQ

(\PSC.SPACEDISP
  [LAMBDA (STREAM WIDTH)                                         ; Edited 28-Sep-93 13:50 by jds
    (POSTSCRIPT.PUTCOMMAND STREAM (\PSC.SPACEWID (DSPFONT NIL STREAM)
                                                 WIDTH)
           " 0 rmoveto "])

(\PSC.SPACEWID
  [LAMBDA (FONTDESC CHAR)                                        ; Edited 28-Sep-93 13:41 by jds
    ;; Spacing character with a special width (e.g. M space, thin (1/5-M) space...
    ;; If CHAR is a list, it's (CHARCODE FACTOR), and we return a width of FACTOR * (CHARWIDTH CHARCODE).  Otherwise, we just return the
    ;; width of CHARCODE.
    (COND
       [(LISTP CHAR)
        (FIXR (FTIMES (CADR CHAR)
                      (CHARWIDTH (CHARCODE.DECODE (CAR CHAR))
                             FONTDESC]
       (T (CHARWIDTH (CHARCODE.DECODE CHAR)
                FONTDESC])

(\PSC.SYMBOLS
  [LAMBDA (STREAM CHAR)                                          ; Edited  2-Nov-94 17:01 by jds
    (LET ((IMAGEDATA (fetch (STREAM IMAGEDATA) of STREAM))
          (OLDFONT (\DSPFONT.PSC STREAM)))
         (\DSPFONT.PSC STREAM (LIST 'ZAPFDINGBATS (fetch (FONTDESCRIPTOR FONTSIZE) of OLDFONT)
                                    (fetch (FONTDESCRIPTOR FONTFACE) of OLDFONT)))
         (POSTSCRIPT.SHOWACCUM STREAM)
         (\UPDATE.PSC STREAM IMAGEDATA)
         (COND
            ((EQUAL CHAR "0,161")
             (POSTSCRIPT.OUTSTR STREAM " bboxchk ")))
         (\DSPFONT.PSC STREAM OLDFONT])

)

;; The mapping of NS characters to Postscript renderings, both as an AList and as a hashtable
```

```
(DEFINEQ

(\POSTSCRIPT.NSHASH
  [LAMBDA (MAPPING-LIST)                                    ; Edited 30-Jul-93 14:46 by sybalskY:MV:ENVOS
                                                            ; Edited  4-May-93 02:21 by jds
                                                            ; Edited  3-Feb-93 00:33 by jds
    (for MAPPING in MAPPING-LIST unless (EQ (CAR MAPPING)
                                            '*)
       do                                                   ; Skip comments in the mapping list.
          (LET [(CHARCODE (CHARCODE.DECODE (CAR MAPPING]
               ;; Fill in the translation entry for this character:
               (PUTHASH CHARCODE [DESTRUCTURING-BIND
                                  (KIND CODE2 BASECHAR UNDERACCENTS)
                                  (SETQ MAPPING (CDR MAPPING))
                                  (CONS KIND (SELECTQ KIND
                                                 ((SYMBOL NIL DINGBAT)
                                                      (CONS (CHARCODE.DECODE CODE2)))
                                                 (FUNCTION (CONS CODE2))
                                                 ((ACCENT ACCENTPAIR)
                                                      (LIST (CHARCODE.DECODE CODE2)
                                                            (CHARCODE.DECODE BASECHAR)
                                                            (AND UNDERACCENTS (CHARCODE.DECODE UNDERACCENTS))))
                                                 (APPLY*        ; Apply setup function to coerce argument data
                                                     ;; MAPPING is of the form ('APPLY* DATA PRINTFN WIDTHFN SETUPFN)  PRINTFN gets applied
                                                     ;; to stream and result of applying SETUPFN to DATA.  WIDTHFN gets applied to coerced data and
                                                     ;; fontdescriptor
                                                         (LIST (APPLY* (OR (CAR (CDDDDR MAPPING))
                                                                           (FUNCTION CL:IDENTITY))
                                                                       (CADR MAPPING))
                                                               (CADDR MAPPING)
                                                               (CADDDR MAPPING)))
                                                 (ERROR "UNRECOGNIZED POSTSCRIPT CHARACTER TYPE" MAPPING]
                                    *POSTSCRIPT-NS-HASH*)
               ;; If this character is in the lower 127, we need to mark it for special handling in \POSTSCRIPT.CHARTYPE, by putting a T in the
               ;; array at the charcode's position:
               (CL:WHEN (<= CHARCODE 254)
                   (CL:SETF (CL:AREF \POSTSCRIPT.CHARTYPE CHARCODE)
                          T))])

)

(RPAQQ *POSTSCRIPT-UNACCENTED-FONTS* (Dancer ZapfDingbats "Dancer" "ZapfDingbats"))

(RPAQQ *POSTSCRIPT-NS-TRANSLATIONS*
       (;; Mapping of NS characters to Postscript renderings.
        ;; First few are for control-codes in old Press fonts (Timesroman, etc.); not strictly NS, but undefined therein so should be OK.
        ("^S" NIL "2,320")                                  ; pressfont em dash
        ("^V" NIL "2,261")                                  ; pressfont en dash
        ("^G" NIL "0,140")
        ("0,244" NIL "2,250")                               ; generic currency symbol
        ("0,251" NIL "2,140")                               ; left single quote
        ("0,254" SYMBOL "2,254")                            ; left arrow
        ("0,255" SYMBOL "2,255")                            ; uparrow
        ("0,256" SYMBOL "2,256")                            ; right arrow
        ("0,257" SYMBOL "2,257")                            ; down arrow
        ("0,260" SYMBOL "2,260")                            ; degree
        ("0,261" SYMBOL "2,261")                            ; +/-
        ("0,264" SYMBOL "2,264")                            ; times
        ("0,267" NIL "2,264")                               ; Center-dot
        ("0,270" SYMBOL "2,270")                            ; divide
        ("0,271" NIL "2,047")                               ; right single quote
        ("0,274" FUNCTION " f14 ")                          ; 1/4
        ("0,275" FUNCTION " f12 ")                          ; 1/2
        ("0,276" FUNCTION " f34 ")                          ; 3/4
        ("0,322" SYMBOL "2,342")                            ; registered
```

```
        ("0,323" SYMBOL "2,343")                                                    ; copyright
        ("0,324" SYMBOL "2,344")                                                    ; tm
        ("0,334" FUNCTION " f18 ")                                                  ; 1/8
        ("0,335" FUNCTION " f38 ")                                                  ; 3/8
        ("0,336" FUNCTION " f58 ")                                                  ; 5/8
        ("0,337" FUNCTION " f78 ")                                                  ; 7/8
        ("0,342" NIL "2,235")                                                       ; Eth (slashed D?)
        ("0,354" NIL "2,237")                                                       ; Thorn
        ("0,363" NIL "2,236")                                                       ; eth
        ("0,374" NIL "2,240")                                                       ; thorn
        ("41,172" DINGBAT "0,110")                                                  ; filled star
        ("42,42" DINGBAT "0,161")                                                   ; ballot-box
        ("42,61" APPLY* "0,161" \PSC.SYMBOLS \PSC.SPACEWID NIL)                     ; Checked ballot-box
        ("357,44" NIL "2,261")                                                      ; n dash
        ("357,45" NIL "2,320")                                                      ; m dash
        ("357,55" APPLY* "M" \PSC.SPACEDISP \PSC.SPACEWID NIL)                      ; M quad
        ("357,54" APPLY* "N" \PSC.SPACEDISP \PSC.SPACEWID NIL)                      ; N quad
        ("357,56" APPLY* "1" \PSC.SPACEDISP \PSC.SPACEWID NIL)                      ; FIGURE quad
        ("357,57" APPLY* ("M" 0.2)
               \PSC.SPACEDISP \PSC.SPACEWID NIL)                                    ; This space (1/5M)
        ("357,60" NIL "2,262")                                                      ; dagger
        ("357,61" NIL "2,263")                                                      ; double dagger
        ("357,062" SYMBOL "2,361")                                                  ; angleright
        ("357,063" SYMBOL "2,341")                                                  ; angleleft
        ("357,70" SYMBOL "2,315")                                                   ; perpendicular
        ("357,101" NIL "2,275")                                                     ; per mil o/oo
        ("357,104" ACCENTPAIR "<" NIL "/")                                          ; not less than
        ("357,105" ACCENTPAIR ">" "/")                                              ; not greater than
        ("357,110" SYMBOL "2,312")                                                  ; parallel
        ("357,111" SYMBOL "2,315")                                                  ; not parallel
        ("357,112" SYMBOL "2,316")                                                  ; element
        ("357,113" SYMBOL "2,317")                                                  ; notelement
        ("357,114" SYMBOL "2,047")                                                  ; suchthat
        ("357,115" SYMBOL "2,334")                                                  ; implied by, double arrow left
        ("357,116" SYMBOL "2,333")                                                  ; iff, double arrow
        ("357,117" SYMBOL "2,336")                                                  ; implies, double arrow right
        ("357,120" SYMBOL "2,253")                                                  ; double arrow
        ("357,121" SYMBOL "2,333")                                                  ; double arrow
        ("357,122" SYMBOL "2,333")                                                  ; l/r arrow
        ("357,126" SYMBOL "2,307")                                                  ; intersection
        ("357,127" SYMBOL "2,310")                                                  ; union
        ("357,130" SYMBOL "2,312")                                                  ; reflexsuperset
        ("357,131" SYMBOL "2,315")                                                  ; reflexsubset
        ("357,132" SYMBOL "2,311")                                                  ; propersuperset
```

```
        ("357,133" SYMBOL "2,314")                                  ; propersubset
        ("357,137" SYMBOL "2,313")                                  ; notsubset
        ("357,141" SYMBOL "2,306")                                  ; emptyset
        ("357,142" SYMBOL "2,305")                                  ; circleplus
        ("357,144" SYMBOL "2,304")                                  ; circlemultiply
        ("357,146" NIL "2,267")                                     ; bullet
        ("357,147" SYMBOL "2,260")                                  ; center circle (composition), lowered degree
        ("357,152" SYMBOL "2,330")                                  ; logicalnot
        ("357,154" SYMBOL "2,320")                                  ; angle
        ("357,160" SYMBOL "2,136")                                  ; perpendicular
        ("357,161" SYMBOL "2,265")                                  ; proportional
        ("357,162" SYMBOL "2,272")                                  ; equivalence
        ("357,165" SYMBOL "2,362")                                  ; integral
        ("357,167" SYMBOL "2,273")                                  ; approxequal
        ("357,170" SYMBOL "2,100")                                  ; congruent
        ("357,172" SYMBOL "2,345")                                  ; summation
        ("357,173" SYMBOL "2,325")                                  ; product
        ("357,174" SYMBOL "2,326")                                  ; radical
        ("357,242" SYMBOL "2,246")                                  ; florin
        ("357,260" SYMBOL "2,351")                                  ; Ceiling, left
        ("357,261" SYMBOL "2,371")                                  ; Ceiling, right
        ("357,262" SYMBOL "2,353")                                  ; Floor, left
        ("357,263" SYMBOL "2,373")                                  ; Floor, right
        ("357,264" SYMBOL "2,44")                                   ; exists
        ("357,265" SYMBOL "2,42")                                   ; forall
        ("357,266" SYMBOL "2,331")                                  ; logicaland
        ("357,267" SYMBOL "2,332")                                  ; logicalor
        ("357,271" SYMBOL "2,321")                                  ; gradient
        ("357,272" SYMBOL "2,266")                                  ; partialdiff
        ("357,313" SYMBOL "2,252")                                  ; spade
        ("357,317" DINGBAT "0,63")                                  ; check
        ("357,375" FUNCTION " f13 ")                                ; 1/3
        ("357,376" FUNCTION " f23 ")                                ; 2/3

        ("361,041" ACCENT "0,4" A)
        ("361,042" ACCENT "0,1" A)
        ("361,043" ACCENT "0,2" A)
        ("361,044" ACCENT "0,6" A)
        ("361,045" ACCENTPAIR A "0,305")                            ; A-macron
        ("361,046" ACCENTPAIR A "0,306")                            ; A-breve

        ("361,047" ACCENT "0,3" A)
        ("361,050" ACCENT "0,5" A)
        ("361,055" ACCENT "0,7" C)
        ("361,060" ACCENT "0,13" E)
        ("361,061" ACCENT "0,10" E)
        ("361,062" ACCENT "0,11" E)
        ("361,063" ACCENTPAIR E "0,305")                            ; E-macron
        ("361,065" ACCENT "0,12" E)
        ("361,066" ACCENTPAIR E NIL "0,316")                        ; E-ogonek

        ("361,076" ACCENT "0,17" I)
        ("361,077" ACCENT "0,14" I)
```

```
        ("361,100" ACCENT "0,15" I)
        ("361,102" ACCENTPAIR I "0,305")
                                                        ; I-macron
        ("361,104" ACCENT "0,16" I)
        ("361,114" ACCENT "0,20" N)
        ("361,117" ACCENT "0,24" O)
        ("361,120" ACCENT "0,21" O)
        ("361,121" ACCENT "0,22" O)
        ("361,122" ACCENT "0,25" O)
        ("361,123" ACCENTPAIR O "0,305")
                                                        ; O-macron
        ("361,124" ACCENT "0,23" O)
        ("361,134" ACCENT "0,26" S)
        ("361,137" ACCENT "0,32" U)
        ("361,140" ACCENT "0,27" U)
        ("361,141" ACCENT "0,30" U)
        ("361,143" ACCENTPAIR U "0,305")
                                                        ; U-macron
        ("361,145" ACCENT "0,31" U)
        ("361,155" ACCENT "0,33" Y)
        ("361,160" ACCENT "0,34" Z)
        ("361,165" ACCENTPAIR Y "0,305")
                                                        ; Y-macron
        ("361,166" ACCENTPAIR "0,341" "0,305")
                                                        ; AE-macron
        ("361,167" ACCENTPAIR "0,352" "0,305")
                                                        ; OE-macron
        ("361,241" ACCENT "0,204" a)
        ("361,242" ACCENT "0,201" a)
        ("361,243" ACCENT "0,202" a)
        ("361,244" ACCENT "0,206" a)
        ("361,245" ACCENTPAIR a "0,305")
                                                        ; a-macron
        ("361,246" ACCENTPAIR a "0,306")
                                                        ; a-breve
        ("361,247" ACCENT "0,203" a)
        ("361,250" ACCENT "0,205" a)
        ("361,255" ACCENT "0,207" c)
        ("361,260" ACCENT "0,213" e)
        ("361,261" ACCENT "0,210" e)
        ("361,262" ACCENT "0,211" e)
        ("361,263" ACCENTPAIR e "0,305")
                                                        ; e-macron
        ("361,265" ACCENT "0,212" e)
        ("361,266" ACCENTPAIR e NIL "0,316")
                                                        ; e-ogonek
        ("361,267" ACCENTPAIR e "0,317")
                                                        ; e-caron
        ("361,276" ACCENT "0,217" i)
        ("361,277" ACCENT "0,214" i)
        ("361,300" ACCENT "0,215" i)
        ("361,302" ACCENTPAIR "0,365" "0,305")
                                                        ; i-macron
        ("361,304" ACCENT "0,216" i)
        ("361,314" ACCENT "0,220" n)
        ("361,317" ACCENT "0,224" o)
        ("361,320" ACCENT "0,221" o)
        ("361,321" ACCENT "0,222" o)
        ("361,322" ACCENT "0,225" o)
        ("361,323" ACCENTPAIR o "0,305")
                                                        ; o-macron
        ("361,324" ACCENT "0,223" o)
        ("361,334" ACCENT "0,226" s)
        ("361,337" ACCENT "0,232" u)
        ("361,340" ACCENT "0,227" u)
        ("361,341" ACCENT "0,230" u)
        ("361,343" ACCENTPAIR u "0,305")
                                                        ; u-macron
        ("361,344" ACCENTPAIR u "0,306")
                                                        ; u-breve
        ("361,345" ACCENT "0,231" u)
        ("361,355" ACCENT "0,233" y)
        ("361,360" ACCENT "0,234" z)
        ("361,365" ACCENTPAIR y "0,305")
                                                        ; y-macron
        ("361,366" ACCENTPAIR "0,361" "0,305")
                                                        ; ae-macron
        ("361,367" ACCENTPAIR "0,372" "0,305")
                                                        ; oe-macron
        ("361,371" ACCENTPAIR a "0,317")
                                                        ; a-caron
        ("361,375" ACCENTPAIR g "0,317")
                                                        ; g-caron

    ;; Special code assignments for Dictionary of Old English, UToronto:
        ("361,370" ACCENTPAIR a ("0,305" "0,306"))
                                                        ; a - breve-macron
```

```
("361,372" ACCENTPAIR e "0,306")                    ; e-breve
("361,373" ACCENTPAIR e "0,305" "0,56")             ; e macron underdot
("361,374" ACCENTPAIR e ("0,305" "0,306"))          ; e - breve-macron
("361,376" ACCENTPAIR "0,365" "0,306")              ; i-breve
("362,242" ACCENTPAIR "0,365" "0,317")              ; i-caron
("362,241" ACCENTPAIR "0,365" ("0,305" "0,306"))    ;  i - breve-macron
("362,243" ACCENTPAIR n "0,305")                    ; n-macron
("362,244" ACCENTPAIR m "0,305")                    ; m-macron
("362,245" ACCENTPAIR o "0,317")                    ; o-caron
("362,246" ACCENTPAIR o "0,306")                    ; o-breve
("362,247" ACCENTPAIR o ("0,305" "0,306"))          ; o - breve-macron
("362,250" ACCENTPAIR o "0,305" "0,56")             ; o-macron underdot
("362,251" ACCENTPAIR o "0,316")                    ; o-ogonek
("362,252" ACCENTPAIR u "0,317")                    ; u-caron
("362,253" ACCENTPAIR u ("0,305" "0,306"))          ; u - breve-macron
("362,254" ACCENTPAIR y "0,306")                    ; y-breve
("362,256" ACCENTPAIR y "0,317")                    ; y-caron
("362,255" ACCENTPAIR y ("0,305" "0,306"))          ; y - breve-macron
                                                    ; 235 = Eth
                                                    ; 236 = eth
                                                    ; 237 = Thorn
                                                    ; 240 = thorn

;; NS Greek characters
("46,101" SYMBOL "2,101")                           ; Alpha
("46,102" SYMBOL "2,102")                           ; Beta
("46,103" SYMBOL 0)                                 ; --empty--
("46,104" SYMBOL "2,107")                           ; Gamma
("46,105" SYMBOL "2,104")                           ; Delta
("46,106" SYMBOL "2,105")                           ; Epsilon
("46,107" SYMBOL 0)                                 ; Stigma
("46,110" SYMBOL 0)                                 ; Digamma
("46,111" SYMBOL "2,132")                           ; Zeta
("46,112" SYMBOL "2,110")                           ; Eta
("46,113" SYMBOL "2,121")                           ; Theta
("46,114" SYMBOL "2,111")                           ; Iota
("46,115" SYMBOL "2,113")                           ; Kappa
("46,116" SYMBOL "2,114")                           ; Lambda
("46,117" SYMBOL "2,115")                           ; Mu
("46,120" SYMBOL "2,116")                           ; Nu
("46,121" SYMBOL "2,130")                           ; Xi
("46,122" SYMBOL "2,117")                           ; Omicron
("46,123" SYMBOL "2,120")                           ; Pi
("46,124" SYMBOL 0)                                 ; Koppa
("46,125" SYMBOL "2,122")                           ; Rho
("46,126" SYMBOL "2,123")                           ; Sigma
("46,127" SYMBOL 0)
```

```
        ("46,130" SYMBOL "2,124")                    ; --empty--
        ("46,131" SYMBOL "2,125")                    ; Tau
        ("46,132" SYMBOL "2,106")                    ; Upsilon
        ("46,133" SYMBOL "2,103")                    ; Phi
        ("46,134" SYMBOL "2,131")                    ; Chi
        ("46,135" SYMBOL "2,132")                    ; Psi
        ("46,141" SYMBOL "2,141")                    ; Omega
        ("46,142" SYMBOL "2,142")                    ; alpha
        ("46,143" SYMBOL 0)                          ; beta
        ("46,144" SYMBOL "2,147")                    ; (md beta)
        ("46,145" SYMBOL "2,144")                    ; gamma
        ("46,146" SYMBOL "2,145")                    ; delta
        ("46,147" SYMBOL "2,126")                    ; epsilon
        ("46,150" SYMBOL 0)                          ; stigma
        ("46,151" SYMBOL "2,172")                    ; digamma
        ("46,152" SYMBOL "2,150")                    ; zeta
        ("46,153" SYMBOL "2,161")                    ; eta
        ("46,154" SYMBOL "2,151")                    ; theta
        ("46,155" SYMBOL "2,153")                    ; iota
        ("46,156" SYMBOL "2,154")                    ; kappa
        ("46,157" SYMBOL "2,155")                    ; lambda
        ("46,160" SYMBOL "2,156")                    ; mu
        ("46,161" SYMBOL "2,170")                    ; nu
        ("46,162" SYMBOL "2,157")                    ; xi
        ("46,163" SYMBOL "2,160")                    ; omicron
        ("46,164" SYMBOL 0)                          ; pi
        ("46,165" SYMBOL "2,162")                    ; (koppa)
        ("46,166" SYMBOL "2,163")                    ; rho
        ("46,167" SYMBOL "2,126")                    ; sigma
        ("46,170" SYMBOL "2,164")                    ; (fl sigma)
        ("46,171" SYMBOL "2.165")                    ; tau
        ("46,172" SYMBOL "2,146")                    ; upsilon
        ("46,173" SYMBOL "2,143")                    ; phi
        ("46,174" SYMBOL "2,171")                    ; chi
        ("46,175" SYMBOL "2,167")                    ; psi
                                                     ; omega
    ;; NS Miscellaneous symbols
        ("041,142" SYMBOL "2,271")
                                                     ; notequal
        ("041,145" SYMBOL "2,243")                   ; lessequal
        ("041,146" SYMBOL "2,263")                   ; greaterequal
        ("041,147" SYMBOL "2,245")                   ; infinity
        ("041,150" SYMBOL "2,134")                   ; therefore
        ("041,155" SYMBOL "2,262")                   ; second
        ("356,055" SYMBOL "2,055")                   ; minus
```

```
        ("356,106" SYMBOL "2,340")                                  ; lozenge
        ("356,163" SYMBOL "2,351")                                  ; topleftbracket
        ("356,164" SYMBOL "2,353")                                  ; bottomleftbracket
        ("356,165" SYMBOL "2,352")                                  ; centerbracket
        ("356,166" SYMBOL "2,371")                                  ; toprightbracket
        ("356,167" SYMBOL "2,373")                                  ; bottomrightbracket
        ("356,176" SYMBOL "2,176")                                  ; similar
        ("356,314" SYMBOL "2,251")                                  ; heart
        ("356,340" SYMBOL "2,374")                                  ; toprightbracce
        ("356,341" SYMBOL "2,357")                                  ; braceextend
        ("356,342" SYMBOL "2,375")                                  ; centerrightbracce
        ("356,343" SYMBOL "2,376")                                  ; bottomrightbracce
        ("356,344" SYMBOL "2,354")                                  ; topleftbracce
        ("356,345" SYMBOL "2,356")                                  ; bottomleftbracce
        ("356,346" SYMBOL "2,355")                                  ; centerleftbracce
        ("356,355" SYMBOL "2,363")                                  ; integraltop
        ("356,356" SYMBOL "2,365")                                  ; integralbottom
        ("356,357" SYMBOL "2,364")                                  ; integralcenter
        ))

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS *POSTSCRIPT-NS-HASH*)
)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS \POSTSCRIPT.FRACTION MACRO ((STREAM STRING)
                                      ;; Handle printing of a fraction, given a string that's the name of the PS function (defined in
                                      ;; \POSTSCRIPT.JOB.SETUP) that prints it.  You must put spaces around the name.
                                      (POSTSCRIPT.SHOWACCUM STREAM)
                                      [COND
                                         ((IGREATERP (SETQ NEWXPOS (IPLUS XPOS CHARWID))
                                                 (ffetch POSTSCRIPTRIGHTMARGIN of IMAGEDATA))
                                          (\TERPRI.PSC STREAM)
                                          (SETQ NEWXPOS (IPLUS (ffetch POSTSCRIPTX of IMAGEDATA)
                                                               CHARWID]
                                      [COND
                                         ((NOT (ffetch POSTSCRIPTCHARSTOSHOW of IMAGEDATA))
                                          (COND
                                             ((ffetch POSTSCRIPTPENDINGXFORM of IMAGEDATA)
                                              (\SETXFORM.PSC STREAM IMAGEDATA)))
                                          (COND
                                             ((ffetch POSTSCRIPTFONTCHANGEDFLG of IMAGEDATA)
                                                               ; If font was changed then switch before printing
                                              (\SWITCHFONTS.PSC STREAM IMAGEDATA)))
                                          (COND
                                             ((ffetch POSTSCRIPTMOVEFLG of IMAGEDATA)
                                                               ; likewise for position
                                              (\SETPOS.PSC STREAM IMAGEDATA]
                                      (POSTSCRIPT.OUTSTR STREAM STRING)))
)
)

(RPAQ \POSTSCRIPT.ORIENTATION.MENU
      (create MENU ITEMS _ '(("Landscape" T "Print this file/document/image in Landscape Orientation")
                             ("Portrait" 'NIL "Print this file/document/image in Portrait Orientation"))
              TITLE _ "Orientation" CENTERFLG _ T MENUOFFSET _ (create POSITION XCOORD _ -1 YCOORD _ 0)
              CHANGEOFFSETFLG _ 'Y))

(RPAQ \POSTSCRIPT.ORIENTATION.OPTIONS.MENU (create MENU ITEMS _ '(("Ask" 'ASK "Always ask whether to print in
                                                                      Landscape or Portrait Orientation")
                                                                 ("Landscape" T "Default printing to Landscape
                                                                      Orientation")
                                                                 ("Portrait" 'NIL "Default printing to Portrait
                                                                      Orientation"))
```

```
                                        TITLE _ "Default Orientation" CENTERFLG _ T))

(RPAQ PS.BITMAPARRAY (READARRAY-FROM-LIST 16 'BYTE 0
                          '(48 49 50 51 52 53 54 55 56 57 65 66 67 68 69 70 NIL)))

(RPAQQ \POSTSCRIPT.JOB.SETUP
       ("/bdef {bind def} bind def" "/ldef {load def} bdef" "/S /show ldef" "/M /moveto ldef" "/DR {transform
              round exch round exch itransform} bdef" "/L {gsave newpath setlinewidth 0 setlinecap" "  M lineto
              currentpoint stroke grestore M} bdef" "/L1 {gsave newpath 0 setdash setgray setlinewidth 0
              setlinecap" "  M lineto currentpoint stroke grestore M} bdef" "/F {findfont exch scalefont
              setfont} bdef" "/CLP {newpath M dup 0 rlineto exch 0 exch rlineto" "  neg 0 rlineto closepath clip
              newpath} bdef" "/R {gsave setgray newpath M dup 0 rlineto exch 0 exch" "  rlineto neg 0 rlineto
              closepath eofill grestore} bdef" "/ellipsedict 9 dict def" "ellipsedict /mtrx matrix put"
              "/ellipse" " { ellipsedict begin" "  /endangle exch def" "  /startangle exch def" "  /orientation
              exch def" "  /minorrad exch def" "  /majorrad exch def" "  /y exch def" "  /x exch def" "
              /savematrix mtrx currentmatrix def" "  x y translate" "  orientation rotate" "  majorrad minorrad
              scale" "  0 0 1 startangle endangle arc" "  savematrix setmatrix" " end } bdef" "/concatprocs"
              " {/proc2 exch cvlit def" "  /proc1 exch cvlit def" "  /newproc proc1 length proc2 length add
              array def" "  newproc 0 proc1 putinterval" "  newproc proc1 length proc2 putinterval" "  newproc
              cvx" " } bdef" "/resmatrix matrix def" "/findresolution" " {72 0 resmatrix defaultmatrix
              dtransform" "  /yres exch def /xres exch def" "  xres dup mul yres dup mul add sqrt" " } bdef"
              "/thebitimage" " {/maskp exch def" "  /bihgt exch def" "  /biwid exch def" "  /byte 1 string def"
              "  /strbufl biwid 8 div ceiling cvi def" "  /strbuf strbufl string def" "  maskp not{{1 exch sub}
              currenttransfer concatprocs settransfer} if" "  biwid bihgt" "  maskp { true } { 1 } ifelse" "
              [biwid 0 0 bihgt 0 0]" "  {/col 0 def" "   {currentfile byte readhexstring pop 0 get" "    dup
              16#B2 eq {pop" "    currentfile byte readhexstring pop 0 get 1 add" "    currentfile byte
              readhexstring pop pop /nbyte byte 0 get def" "    { strbuf col nbyte put /col col 1 add def}
              repeat}" "   {dup 16#B3 eq {pop /col col" "    currentfile byte readhexstring pop" "    0 get add
              1 add def}" "    {16#B4 eq {currentfile byte readhexstring pop pop} if" "     strbuf col byte 0
              get put /col col 1 add def} ifelse" "   } ifelse" "   col strbufl ge { exit } if } loop" "
              strbuf }" "  maskp { imagemask } { image } ifelse" " } bdef" "/setuserscreendict 22 dict def"
              "setuserscreendict begin" " /tempctm matrix def" " /temprot matrix def" " /tempscale matrix def"
              "end" "/setuserscreen" " {setuserscreendict begin" "   /spotfunction exch def" "   /screenangle
              exch def" "   /cellsize exch def" "   /m tempctm currentmatrix def" "   /rm screenangle temprot
              rotate def" "   /sm cellsize dup tempscale scale def" "   sm rm m m concatmatrix m concatmatrix
              pop" "   1 0 m dtransform /y1 exch def /x1 exch def" "   /veclength x1 dup mul y1 dup mul add sqrt
              def" "   /frequency findresolution veclength div def" "   /newscreenangle y1 x1 atan def" "   m 2
              get m 1 get mul m 0 get m 3 get mul sub" "   0 gt { { neg } /spotfunction load concatprocs" "
              /spotfunction exch def } if" "   frequency newscreenangle /spotfunction load setscreen" "  end"
              " } bdef" "/setpatterndict 18 dict def" "setpatterndict begin" " /bitison" "  {/ybit exch def
              /xbit exch def" "   /bytevalue bstring ybit bwidth mul xbit 8 idiv add get def" "   /mask 1 7 xbit
              8 mod sub bitshift def" "   bytevalue mask and 0 ne" "  } bdef" "end" "/bitpatternspotfunction"
              " {setpatterndict begin" "   /y exch def /x exch def" "   /xindex x 1 add 2 div bpside mul 1 sub
              cvi def" "   /yindex y 1 add 2 div bpside mul 1 sub cvi def" "   xindex yindex bitison" "
              {/onbits onbits 1 add def 1}" "    {/offbits offbits 1 add def 0} ifelse" "  end" " } bdef"
              "/setpattern" " {setpatterndict begin" "   /cellsz exch def" "   /angle exch def" "   /bwidth exch
              def" "   /bpside exch def" "   /bstring exch def" "   /onbits 0 def /offbits 0 def" "   cellsz
              angle /bitpatternspotfunction load setuserscreen" "   {} settransfer" "   offbits offbits onbits
              add div setgray" "  end" " } bdef" "%% - - - - - Fraction-setting code, to support NS fonts better
              - - - - -" "/fractiondict 20 dict def" "/fractionshow " "{ fractiondict begin" "/denom exch def "
              "/num exch def " "/regfont currentfont def" "/fractfont currentfont [.65 0 0 .6 0 0] makefont def
" "gsave newpath 0 0 moveto " "(1) true charpath flattenpath pathbbox " "/height exch def pop pop pop" "
              grestore" "0 .4 height mul rmoveto" "fractfont setfont num show" "0 .4 height mul neg rmoveto
              regfont setfont (\244) show" "fractfont setfont denom show regfont setfont end } bdef" "/f14 { (1)
              (4) fractionshow } bdef" "/f12 { (1) (2) fractionshow } bdef" "/f34 { (3) (4) fractionshow }
              bdef" "/f18 { (1) (8) fractionshow } bdef" "/f38 { (3) (8) fractionshow } bdef" "/f58 { (5) (8)
              fractionshow } bdef" "/f78 { (7) (8) fractionshow } bdef" "/f13 { (1) (3) fractionshow } bdef"
              "/f23 { (2) (3) fractionshow } bdef" "/bboxdict 20 dict def" "/bboxchk { bboxdict begin"
              "/regfont currentfont def" "/chkfont currentfont [1.25 0 0 1.25 0 0] makefont def " "gsave newpath
              0 0 moveto " "(\161) true charpath flattenpath pathbbox " "/height exch def pop pop pop " "
              grestore " " currentpoint " " .2 height mul .3 height mul rmoveto" "chkfont setfont (\063) show"
              " moveto" " regfont setfont" "(\161) show end } bdef" "/rencdict 15 dict def" "/encodefont {
              rencdict begin" "/newname exch def" "/oldfont exch def" "/newcodes [" "8#001 /Aacute" "8#002
              /Acircumflex" "8#003 /Adieresis" "8#004 /Agrave" "8#005 /Aring" "8#006 /Atilde" "8#007 /Ccedilla"
              "8#010 /Eacute" "8#011 /Ecircumflex" "8#012 /Edieresis" "8#013 /Egrave" "8#014 /Iacute" "8#015
              /Icircumflex" "8#016 /Idieresis" "8#017 /Igrave" "8#020 /Ntilde" "8#021 /Oacute" "8#022
              /Ocircumflex" "8#023 /Odieresis" "8#024 /Ograve" "8#025 /Otilde" "8#026 /Scaron" "8#027 /Uacute"
              "8#030 /Ucircumflex" "8#031 /Udieresis" "8#032 /Ugrave" "8#033 /Ydieresis" "8#034 /Zcaron"
              "8#177 /periodinferior" "8#201 /aacute" "8#202 /acircumflex" "8#203 /adieresis" "8#204 /agrave"
              "8#205 /aring" "8#206 /atilde" "8#207 /ccedilla" "8#210 /eacute" "8#211 /ecircumflex" "8#212
              /edieresis" "8#213 /egrave" "8#214 /iacute" "8#215 /icircumflex" "8#216 /idieresis" "8#217
              /igrave" "8#220 /ntilde" "8#221 /oacute" "8#222 /ocircumflex" "8#223 /odieresis" "8#224 /ograve"
              "8#225 /otilde" "8#226 /scaron" "8#227 /uacute" "8#230 /ucircumflex" "8#231 /udieresis" "8#232
              /ugrave" "8#233 /ydieresis" "8#234 /zcaron" "8#235 /Eth" "8#236 /eth" "8#237 /Thorn" "8#240
              /thorn" " ] def" "/olddict oldfont findfont def /newfont olddict maxlength dict def" "olddict {
              exch dup /FID ne { dup /Encoding eq" "{ exch dup length array copy newfont 3 1 roll put }" "{ exch
              newfont 3 1 roll put } ifelse }" " { pop pop } ifelse } forall" "newfont /FontName newname put"
              "newcodes aload pop" "newcodes length 2 idiv { newfont /Encoding get 3 1 roll put } repeat "
              "newname newfont definefont pop end } def" " /accentdict 10 dict def " " /accentor { accentdict
              begin /scaler exch def /delta exch def " "/unders exch def /accents exch def /mainch exch def
              /scrt (X) def" " /w1 mainch stringwidth pop def " " currentpoint mainch show currentpoint 4 2 roll
" "accents { /ch exch def 2 copy moveto " "         scrt 0 ch put " "         /w2 scrt stringwidth pop def "
              "         w1 w2 sub 2 div delta rmoveto scrt show " "         /delta delta 150 scaler mul 9 div
              add def" "        } forall " "unders { /ch exch def 2 copy moveto " "         scrt 0 ch put " "
              /w2 scrt stringwidth pop def " "         ch 46 eq { w1 w2 sub 2 div -175 scaler mul 9 div rmoveto
              scrt show 0 175 rmoveto }" "          { w1 w2 sub 2 div 0 rmoveto scrt show } ifelse " "        }
              forall " " pop pop moveto end } def " "%%%%EndProlog" "%%%%BeginSetup"))
```

```
(RPAQQ SlopeMenuItems ((Italic 'ITALIC "This is an Italic Slope font")
                       (Regular 'REGULAR "This is a Regular Slope font")))

(RPAQQ WeightMenuItems ((Bold 'BOLD "This is a Bold Weight font")
                        (Medium 'MEDIUM "This is a Medium Weight font")
                        (Light 'LIGHT "This is a Light Weight font")))

(ADDTOVAR BackgroundMenuCommands
          ("PS Orientation" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE (MENU \POSTSCRIPT.ORIENTATION.OPTIONS.MENU))
                 "Select the default Orientation for PostScript output"
                 (SUBITEMS ("Ask" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE 'ASK)
                                  "Always ask whether to print in Landscape or Portrait Orientation")
                        ("Landscape" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE T)
                               "Default printing to Landscape Orientation")
                        ("Portrait" '(SETQ POSTSCRIPT.PREFER.LANDSCAPE NIL)
                               "Default printing to Portrait Orientation"))))

(RPAQQ BackgroundMenu NIL)

(DECLARE%: EVAL@COMPILE

(RPAQQ GOLDEN.RATIO 1.618034)

(RPAQQ \PS.SCALE0 100)

(RPAQQ \PS.TEMPARRAYLEN 20)

(CONSTANTS (GOLDEN.RATIO 1.618034)
       (\PS.SCALE0 100)
       (\PS.TEMPARRAYLEN 20))
)

(RPAQ? POSTSCRIPT.BITMAP.SCALE 1)

(RPAQ? POSTSCRIPT.EOL 'CR)

(RPAQ? POSTSCRIPT.IMAGESIZEFACTOR 1)

(RPAQ? POSTSCRIPT.PREFER.LANDSCAPE NIL)

(RPAQ? POSTSCRIPT.TEXTFILE.LANDSCAPE NIL)

(RPAQ? POSTSCRIPT.DEFAULT.PAGEREGION '(4800 4800 52800 70800))

(RPAQ? POSTSCRIPT.TEXTURE.SCALE 4)

(RPAQ? POSTSCRIPTFONTDIRECTORIES (LIST (COND ((EQ (MACHINETYPE)
                                                  'MAIKO)
                                              "{dsk}/USR/LOCAL/LDE/FONTS/POSTSCRIPT/")
                                             (T "{DSK}<LISPFILES>POSTSCRIPT>"))))

(RPAQ? \POSTSCRIPT.MAX.WILD.FONTSIZE 72)

(DEFINEQ
```

(POSTSCRIPTSEND

```
  [LAMBDA (HOST FILE PRINTOPTIONS)                          ; Edited 20-Nov-95 11:29 by
                                                            ; Edited 20-Nov-95 11:26 by
```

;; This is the send function for generic POSTSCRIPT printers. It branches on the architecture-specific function. The theory is that the send method ;; is really a property of the operating system, not a property of specific postscript printers. These functions are contained in separate library files ;; (or defined by user).

```
    (SELECTQ (MKATOM (UNIX-GETPARM "ARCH"))
         (dos (DOSPRINT HOST FILE PRINTOPTIONS))
         (UnixPrint HOST FILE PRINTOPTIONS])

)

(ADDTOVAR PRINTERTYPES ((POSTSCRIPT)
                        (CANPRINT (POSTSCRIPT))
                        (STATUS TRUE)
                        (PROPERTIES NILL)
                        (SEND POSTSCRIPTSEND)
                        (BITMAPSCALE POSTSCRIPT.BITMAPSCALE)
                        (BITMAPFILE (POSTSCRIPT.HARDCOPYW FILE BITMAP SCALEFACTOR REGION ROTATION TITLE))))

(ADDTOVAR POSTSCRIPT.FONT.ALIST (HELVETICA . HELVETICA)
                                (HELVETICAD . HELVETICA)
                                (TIMESROMAN . TIMES)
                                (TIMESROMAND . TIMES)
                                (COURIER . COURIER)
                                (GACHA . COURIER)
                                (CLASSIC . NEWCENTURYSCHLBK)
                                (MODERN . HELVETICA)
                                (CREAM . HELVETICA)
                                (TERMINAL . COURIER)
```

```
                                (LOGO . HELVETICA)
                                (OPTIMA . PALATINO)
                                (TITAN . COURIER))

(ADDTOVAR PRINTFILETYPES (POSTSCRIPT (TEST POSTSCRIPTFILEP)
                                (EXTENSION (PS PSC PSF))
                                (CONVERSION (TEXT POSTSCRIPT.TEXT TEDIT POSTSCRIPT.TEDIT))))

(ADDTOVAR IMAGESTREAMTYPES (POSTSCRIPT (OPENSTREAM OPENPOSTSCRIPTSTREAM)
                                    (FONTCREATE POSTSCRIPT.FONTCREATE)
                                    (FONTSAVAILABLE POSTSCRIPT.FONTSAVAILABLE)
                                    (CREATECHARSET \CREATECHARSET.PSC)))

(RPAQ? POSTSCRIPT.PAGETYPE 'LETTER)
```

;; NIL means initial clipping is same as paper size. Don't know why the other regions were specified--rmk

```
(APPENDTOVAR POSTSCRIPT.PAGEREGIONS (LETTER (0 0 8.5 11)
                                            NIL
                                            (-0.1 -0.1 8.7 11.2))
       (LEGAL (0 0 8.5 14)
              NIL
              (-0.1 -0.1 8.7 14.2))
       (NOTE (0 0 8.5 11)
             NIL
             (-0.1 -0.1 8.7 11.2)))

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS DEFAULTPRINTINGHOST POSTSCRIPT.BITMAP.SCALE POSTSCRIPT.EOL POSTSCRIPT.FONT.ALIST
       POSTSCRIPT.PREFER.LANDSCAPE POSTSCRIPT.TEXTFILE.LANDSCAPE POSTSCRIPT.TEXTURE.SCALE
       POSTSCRIPTFONTDIRECTORIES \POSTSCRIPT.JOB.SETUP \POSTSCRIPT.MAX.WILD.FONTSIZE
       \POSTSCRIPT.ORIENTATION.MENU \POSTSCRIPTIMAGEOPS POSTSCRIPT.PAGETYPE POSTSCRIPT.PAGEREGIONS)
)

(DECLARE%: DONTEVAL@LOAD DOCOPY

(POSTSCRIPT.INIT)
)

(PUTPROPS POSTSCRIPTSTREAM FILETYPE :TCOMPL)

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

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA POSTSCRIPT.PUTCOMMAND)
)
```

FUNCTION INDEX

VARIABLE INDEX

RECORD INDEX

CONSTANT INDEX

MACRO INDEX

PROPERTY INDEX