File created: 23-Jul-90 18:28:51 **{DSK}<home>peach>matsuda>NEW-SKETCH-COLOR.;2**

previous date: 23-Jul-90 17:49:19 {DSK}<home>peach>matsuda>NEW-SKETCH-COLOR.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
;; Copyright (c) 1990 by Fuji Xerox Co., Ltd.  All rights reserved.

(RPAQQ NEW-SKETCH-COLORCOMS
       [(P  (MOVD 'CIRCLE.DRAWFN 'ORG.CIRCLE.DRAWFN)
            (MOVD 'CLOSED.WIRE.DRAWFN 'ORG.CLOSED.WIRE.DRAWFN)
            (MOVD 'BOX.DRAWFN1 'ORG.BOX.DRAWFN1)
            (SETQ SKETCHINCOLORFLG T))
        (FNS \BBTCURVEPT BMOBJ.DISPLAYFN BITMAPOBJ.SNAPW OPPOSITECOLOR \SCALEDBITBLT.DISPLAY BITMAPELT.INPUTFN
             GET.BITMAP.POSITION BOX.DRAWFN1 CIRCLE.DRAWFN CLOSED.WIRE.DRAWFN SKETCHINCOLORP NEW.READCOLOR1
             SK.FIGUREIMAGE)
        (P  (MOVD 'READCOLOR1 'ORG.READCOLOR1)
            (MOVD 'NEW.READCOLOR1 'READCOLOR1)
            (REPLACE (IMAGEOPS IMFILLPOLYGON)
                   OF \8DISPLAYIMAGEOPS WITH (FUNCTION POLYSHADE.DISPLAY))
            (REPLACE (IMAGEOPS IMSCALEDBITBLT)
                   OF \8DISPLAYIMAGEOPS WITH (FUNCTION \SCALEDBITBLT.DISPLAY])

(MOVD 'CIRCLE.DRAWFN 'ORG.CIRCLE.DRAWFN)

(MOVD 'CLOSED.WIRE.DRAWFN 'ORG.CLOSED.WIRE.DRAWFN)

(MOVD 'BOX.DRAWFN1 'ORG.BOX.DRAWFN1)

(SETQ SKETCHINCOLORFLG T)

(DEFINEQ

(\BBTCURVEPT
  [LAMBDA (X Y BBT LEFT BRUSHWIDTH LEFTMINUSBRUSH RIGHTPLUS1 NBITSRIGHTPLUS1 TOPMINUSBRUSH DestinationBitMap
             BRUSHHEIGHT BOTTOMMINUSBRUSH TOP BRUSHBASE DESTINATIONBASE RASTERWIDTH BRUSHRASTERWIDTH
             COLORBRUSHBASE NBITS DISPLAYDATA)                    ; Edited 24-May-90 10:03 by matsuda
    ;; Called by \CURVEPT macro.  Draws a brush point by bitblting BRUSHBM to point X,Y in DestinationBitMap.  BBT is a BitBlt table where
    ;; everything is already set except the source and destination addresses, width and height.  In other words, only the easy stuff
                                                                    ; set the width fields of the bbt
    [PROG (CLIPPEDTOP STY)
          [COND
             [(ILEQ Y TOPMINUSBRUSH)                                ; the top part of the brush is visible
              (SETQ CLIPPEDTOP (IPLUS Y BRUSHHEIGHT))
              (replace PBTSOURCE of BBT with BRUSHBASE)
              (freplace PBTHEIGHT of BBT with (IMIN BRUSHHEIGHT (IDIFFERENCE Y BOTTOMMINUSBRUSH]
             (T                                                     ; only the bottom is visible
                (SETQ CLIPPEDTOP TOP)
                [replace PBTSOURCE of BBT with (\ADDBASE BRUSHBASE (ITIMES BRUSHRASTERWIDTH (SETQ STY
                                                                                              (IDIFFERENCE Y
                                                                                                     TOPMINUSBRUSH]
                (freplace PBTHEIGHT of BBT with (IDIFFERENCE (IMIN BRUSHHEIGHT (IDIFFERENCE Y BOTTOMMINUSBRUSH))
                                                             STY]
          (freplace PBTDEST of BBT with (\ADDBASE DESTINATIONBASE (ITIMES RASTERWIDTH (\SFInvert DestinationBitMap
                                                                                        CLIPPEDTOP]
    [COND
       (COLORBRUSHBASE [COND
                          [(ILESSP X LEFT)                          ; only the right part of the brush is visible
                                                                    ; FOR NOW BRUTE FORCE WITH NBITS CHECK
                           [freplace PBTDESTBIT of BBT with (COND
                                                              ((EQ NBITS 4)
                                                               (LLSH LEFT 2))
                                                              (T (LLSH LEFT 3]
                           (freplace PBTSOURCEBIT of BBT with (IDIFFERENCE BRUSHWIDTH
                                                                (freplace PBTWIDTH of BBT
                                                                   with (COND
                                                                          ((EQ NBITS 4)
                                                                           (LLSH (IDIFFERENCE X
                                                                                              LEFTMINUSBRUSH)
                                                                                 2))
                                                                          (T (LLSH (IDIFFERENCE X
                                                                                                LEFTMINUSBRUSH)
                                                                                   3]
                          (T                                        ; left edge is visible
                             [freplace PBTDESTBIT of BBT with (SETQ X (COND
                                                                        ((EQ NBITS 4)
                                                                         (LLSH X 2))
                                                                        (T (LLSH X 3]
                             (freplace PBTSOURCEBIT of BBT with 0)  ; set width to the amount that is visible
                             (freplace PBTWIDTH of BBT with (IMIN BRUSHWIDTH (IDIFFERENCE NBITSRIGHTPLUS1 X]
             (COND
```

```
                ((NEQ (ffetch DDOPERATION of DISPLAYDATA)
                      'INVERT)                                  ; if color brush is used, the ground must be cleared before the
                                                                ; brush is put in.
                 (\SETPBTFUNCTION BBT (ffetch DDSOURCETYPE of DISPLAYDATA)
                        'ERASE)
                 (\PILOTBITBLT BBT 0)                           ; reset the source to point to the color bitmap.

                 ))
              [COND
                 ((ILEQ Y TOPMINUSBRUSH)                        ; the top part of the brush is visible
                  (freplace PBTSOURCE of BBT with COLORBRUSHBASE))
                 (T                                             ; only the bottom is visible
                    (freplace PBTSOURCE of BBT with (\ADDBASE COLORBRUSHBASE (ITIMES BRUSHRASTERWIDTH
                                                                                    (IDIFFERENCE Y TOPMINUSBRUSH]
              (\SETPBTFUNCTION BBT (ffetch DDSOURCETYPE of DISPLAYDATA)
                     (ffetch DDOPERATION of DISPLAYDATA)))
       (T (COND
             [(ILESSP X LEFT)                                   ; only the right part of the brush is visible
              (freplace PBTDESTBIT of BBT with LEFT)
              (freplace PBTSOURCEBIT of BBT with (IDIFFERENCE BRUSHWIDTH (freplace PBTWIDTH of BBT
                                                                            with (IDIFFERENCE X LEFTMINUSBRUSH]
             (T                                                 ; left edge is visible
                (freplace PBTDESTBIT of BBT with X)
                (freplace PBTSOURCEBIT of BBT with 0)           ; set width to the amount that is visible
                (freplace PBTWIDTH of BBT with (IMIN BRUSHWIDTH (IDIFFERENCE RIGHTPLUS1 X]
   (\PILOTBITBLT BBT 0])
```

(**BMOBJ.DISPLAYFN**

```
[LAMBDA (IMAGEOBJ IMAGE.STREAM)                                 ; Edited 18-Apr-90 16:28 by matsuda
```

;; Display a bitmap IMAGEOBJ on IMAGE.STREAM. Scales and rotates it if appropriate, and moves it down by DESCENT.

```
(PROG ([FACTOR (fetch (BITMAPOBJ BMOBJSCALEFACTOR) of (IMAGEOBJPROP IMAGEOBJ 'OBJECTDATUM]
       [BITMAP (fetch (BITMAPOBJ BITMAP) of (IMAGEOBJPROP IMAGEOBJ 'OBJECTDATUM]
       (CACHE (IMAGEOBJPROP IMAGEOBJ 'CACHED.BITMAP))
       [DESCENT (fetch (BITMAPOBJ BMOBJDESCENT) of (IMAGEOBJPROP IMAGEOBJ 'OBJECTDATUM]
       (STREAM-SCALE (DSPSCALE NIL IMAGE.STREAM))
       SHRUNK.BITMAP)
      (RELMOVETO 0 [IMINUS (FIXR (FTIMES STREAM-SCALE (OR DESCENT 0]
            IMAGE.STREAM)
      (SELECTQ (IMAGESTREAMTYPE IMAGE.STREAM)
           (INTERPRESS ;; Printing to an Interpress stream, so use the specialized method.
                       (SHOWBITMAP.IP IMAGE.STREAM BITMAP NIL FACTOR 0))
           ((DISPLAY PRESS) ;; This is the default case, press display and everyone else prints the junky shrunk bitmap
               (COND
                  ((NOT (SETQ SHRUNK.BITMAP CACHE))
                   [COND
                      [(LEQ FACTOR 1.0)                         ; We're shrinking the bitmap.  Create a shrunk image for display
                       (SETQ SHRUNK.BITMAP (SHRINKBITMAP BITMAP (FQUOTIENT 1.0 FACTOR)
                                                         (FQUOTIENT 1.0 FACTOR]
                      (T                                        ; We're expanding it.  Create a bigger one.
                         (SETQ SHRUNK.BITMAP (EXPANDBITMAP BITMAP FACTOR FACTOR]
                   (IMAGEOBJPROP IMAGEOBJ 'CACHED.BITMAP SHRUNK.BITMAP)))
               [BITBLT SHRUNK.BITMAP NIL NIL IMAGE.STREAM (DSPXPOSITION NIL IMAGE.STREAM)
                      (DSPYPOSITION NIL IMAGE.STREAM)
                      (FIXR (FTIMES FACTOR (BITMAPWIDTH BITMAP)))
                      (FIXR (FTIMES FACTOR (BITMAPHEIGHT BITMAP])
           (PROGN ;; This is the default case--Call SCALEDBITBLT
                  (SCALEDBITBLT BITMAP 0 0 IMAGE.STREAM (DSPXPOSITION NIL IMAGE.STREAM)
                         (DSPYPOSITION NIL IMAGE.STREAM)
                         (BITMAPWIDTH BITMAP)
                         (BITMAPHEIGHT BITMAP)
                         'INPUT
                         'PAINT NIL NIL FACTOR])
```

(**BITMAPOBJ.SNAPW**

```
[LAMBDA NIL                                                     ; Edited 12-Apr-90 09:09 by matsuda
```

(* * makes an image object of a prompted for region of the screen.)

```
(PROG ((REG (GETREGION))
       BM)
      [SETQ BM (BITMAPCREATE (fetch (REGION WIDTH) of REG)
                      (fetch (REGION HEIGHT) of REG)
                      (BITSPERPIXEL (SCREENBITMAP \CURSORSCREEN]
      (BITBLT (SCREENBITMAP \CURSORSCREEN)
             (fetch (REGION LEFT) of REG)
             (fetch (REGION BOTTOM) of REG)
             BM 0 0 NIL NIL 'INPUT 'REPLACE)
      (COPYINSERT (BITMAPTEDITOBJ BM 1 0))
      (RETURN])
```

(OPPOSITECOLOR

```
[LAMBDA (COLOR BITSPERPIXEL)                                    ; Edited 23-May-90 15:05 by matsuda
  (IDIFFERENCE (MAXIMUMCOLOR BITSPERPIXEL)
         (COLORNUMBERP COLOR BITSPERPIXEL])
```

(\SCALEDBITBLT.DISPLAY

```
[LAMBDA (SOURCEBITMAP SOURCELEFT SOURCEBOTTOM DESTINATION DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT
          SOURCETYPE OPERATION TEXTURE CLIPPINGREGION CLIPPEDSOURCELEFT CLIPPEDSOURCEBOTTOM SCALE)
                                                                ; Edited 22-May-90 15:02 by matsuda
  (LET (BITMAP REGION)
       (IF (NULL SCALE)
           THEN (SETQ SCALE 1))
       (IF (WINDOWP SOURCEBITMAP)
           THEN (SETQ REGION (DSPCLIPPINGREGION NIL SOURCEBITMAP))
                (IF (NULL WIDTH)
                    THEN (SETQ WIDTH (FETCH (REGION WIDTH) OF REGION)))
                (IF (NULL HEIGHT)
                    THEN (SETQ HEIGHT (FETCH (REGION HEIGHT) OF REGION)))
         ELSEIF (BITMAPP SOURCEBITMAP)
           THEN (IF (NULL WIDTH)
                    THEN (SETQ WIDTH (BITMAPWIDTH SOURCEBITMAP)))
                (IF (NULL HEIGHT)
                    THEN (SETQ HEIGHT (BITMAPHEIGHT SOURCEBITMAP)))
         ELSE (SHOULDNT))
       (SETQ BITMAP (BITMAPCREATE WIDTH HEIGHT (FETCH SCBITSPERPIXEL OF \CURSORSCREEN)))
       (BITBLT SOURCEBITMAP SOURCELEFT SOURCEBOTTOM BITMAP)
       (BITBLT (EXPANDBITMAP BITMAP SCALE SCALE)
              NIL NIL DESTINATION DESTINATIONLEFT DESTINATIONBOTTOM (TIMES WIDTH SCALE)
              (TIMES HEIGHT SCALE)
              SOURCETYPE OPERATION TEXTURE CLIPPINGREGION])
```

(BITMAPELT.INPUTFN

```
[LAMBDA (WINDOW)                                                ; Edited 22-May-90 12:56 by matsuda
                                                                (* gets a region of the screen and makes it a scalable bitmap.)
  (PROG ((REGION (GETREGION 4 4))
         BM POS)
        (OR (REGIONP REGION)
            (RETURN))
        (SETQ BM (BITMAPCREATE (fetch (REGION WIDTH) of REGION)
                               (fetch (REGION HEIGHT) of REGION)
                               (FETCH SCBITSPERPIXEL OF \CURSORSCREEN)))
        (BITBLT (SCREENBITMAP \CURSORSCREEN)
               (fetch (REGION LEFT) of REGION)
               (fetch (REGION BOTTOM) of REGION)
               BM 0 0 (fetch (REGION WIDTH) of REGION)
               (fetch (REGION HEIGHT) of REGION))
        (OR (SETQ POS (GET.BITMAP.POSITION WINDOW BM NIL "Place the bitmap image."))
            (RETURN))
        (RETURN (SK.BITMAP.CREATE BM (SK.MAP.INPUT.PT.TO.GLOBAL POS WINDOW)
                       (VIEWER.SCALE WINDOW])
```

(GET.BITMAP.POSITION

```
[LAMBDA (WINDOW BITMAP OPERATION MSG XOFFSET YOFFSET)           ; Edited 22-May-90 12:53 by matsuda

        (* gets a position by tracking with a bitmap The spec returns is actually
        (ONGRID? position) so that caller can tell whether it was placed on grid or not.)

  (PROG (BUFFER.BITMAP WIDTH HEIGHT)
        (SETQ WIDTH (BITMAPWIDTH BITMAP))
        (SETQ HEIGHT (BITMAPHEIGHT BITMAP))
        (SETQ BUFFER.BITMAP (BITMAPCREATE WIDTH HEIGHT (FETCH SCBITSPERPIXEL OF \CURSORSCREEN)))
        (STATUSPRINT WINDOW "
               " MSG)
        (RETURN (SK.TRACK.BITMAP1 WINDOW BITMAP BUFFER.BITMAP WIDTH HEIGHT (OR OPERATION 'PAINT)
                       XOFFSET YOFFSET])
```

(BOX.DRAWFN1

```
[LAMBDA (REG SIZE WIN WINREG OPERATION DASHING TEXTURE OUTLINECOLOR FILLINGCOLOR)
                                                                ; Edited 25-May-90 14:18 by matsuda
                                                                (* draws a box. Used by both box and text box elements.)
  (COND
     ((OR (NULL WINREG)
          (REGIONSINTERSECTP WINREG REG))
      (COND
         ((AND (SKETCHINCOLORP)
               (OR FILLINGCOLOR TEXTURE))                       (* call the filling routine that does color.)
          (FILLPOLYGON (KNOTS.OF.REGION REG SIZE)
                 FILLINGCOLOR WIN))
          (TEXTURE (DSPFILL REG (COND
                                   ((EQ (DSPOPERATION NIL WIN)
                                        'ERASE)                 (* use black in case the window moved because of texture to
                                                                window alignment bug.)
                                    BLACKSHADE)
```

```
                              (T TEXTURE))
                          (SK.TRANSLATE.MODE OPERATION WIN)
                          WIN))
          (FILLINGCOLOR                                          (* if no texture, use the color.)
                 (DSPFILL REG (TEXTUREOFCOLOR FILLINGCOLOR)
                        OPERATION WIN)))

          (* code to fix white space bug in Interpress. It works but Masters are larger and the one I tried wouldn't print.
          (SELECTQ (IMAGESTREAMTYPE WIN) ((NIL DISPLAY PRESS)
          (* special case DISPLAY for speed and PRESS because rounded corners don't work for large brushes.)
          (SK.DRAWAREABOX (fetch (REGION LEFT) of REG) (fetch (REGION BOTTOM) of REG)
          (fetch (REGION WIDTH) of REG) (fetch (REGION HEIGHT) of REG) SIZE OPERATION WIN DASHING OUTLINECOLOR))
          (PROG ((LFT (fetch (REGION LEFT) of REG)) (BTM (fetch (REGION BOTTOM) of REG))
          (TOP (fetch (REGION TOP) of REG)) (RGHT (fetch (REGION RIGHT) of REG)))
          (DRAWCURVE (LIST (CREATEPOSITION LFT BTM) (CREATEPOSITION LFT TOP)
          (CREATEPOSITION RIGHT TOP) (CREATEPOSITION RIGHT BTM)) T
          (create BRUSH BRUSHSHAPE _ (QUOTE ROUND) BRUSHSIZE _ SIZE BRUSHCOLOR _ OUTLINECOLOR) DASHING
          WIN))))

       (SK.DRAWAREABOX (fetch (REGION LEFT) of REG)
              (fetch (REGION BOTTOM) of REG)
              (fetch (REGION WIDTH) of REG)
              (fetch (REGION HEIGHT) of REG)
              SIZE
              (SK.TRANSLATE.MODE OPERATION WIN)
              WIN DASHING OUTLINECOLOR])
```

(CIRCLE.DRAWFN

```
  [LAMBDA (CIRCLEELT WINDOW REGION)                              ; Edited 25-May-90 15:36 by matsuda
                                                                 (* draws a circle from a circle element.)
    (PROG ((GCIRCLE (fetch (SCREENELT INDIVIDUALGLOBALPART) of CIRCLEELT))
           (LCIRCLE (fetch (SCREENELT LOCALPART) of CIRCLEELT))
           CPOS DASHING FILLING)
          (SETQ CPOS (fetch (LOCALCIRCLE CENTERPOSITION) of LCIRCLE))
          (SETQ DASHING (fetch (LOCALCIRCLE LOCALCIRCLEDASHING) of LCIRCLE))
          (SETQ FILLING (fetch (LOCALCIRCLE LOCALCIRCLEFILLING) of LCIRCLE))
          (COND
             ((fetch (SKFILLING FILLING.COLOR) of FILLING)

          (* if the circle is filled with a color call FILLCIRCLE with both the texture and the color.
          This allows iris to do its thing before textures and colors are merged.)

              (DSPOPERATION (PROG1 (DSPOPERATION (fetch (SKFILLING FILLING.OPERATION) of FILLING)
                                           WINDOW)
                                   (FILLCIRCLE (fetch (POSITION XCOORD) of CPOS)
                                          (fetch (POSITION YCOORD) of CPOS)
                                          (fetch (LOCALCIRCLE RADIUS) of LCIRCLE)
                                          (fetch (SKFILLING FILLING.COLOR) of FILLING)
                                          WINDOW))
                            WINDOW))
             ((fetch (SKFILLING FILLING.TEXTURE) of FILLING)     (* if the circle is filled with texture, call FILLCIRCLE.)
              (DSPOPERATION (PROG1 (DSPOPERATION (fetch (SKFILLING FILLING.OPERATION) of FILLING)
                                           WINDOW)
                                   (FILLCIRCLE (fetch (POSITION XCOORD) of CPOS)
                                          (fetch (POSITION YCOORD) of CPOS)
                                          (fetch (LOCALCIRCLE RADIUS) of LCIRCLE)
                                          (COND
                                             ((EQ (DSPOPERATION NIL WINDOW)
                                                  'ERASE)        (* use black in case the window moved because of texture to
                                                                 window alignment bug.)
                                              BLACKSHADE)
                                             (T (fetch (SKFILLING FILLING.TEXTURE) of FILLING)))
                                          WINDOW))
                            WINDOW)))
          (RETURN (\CIRCLE.DRAWFN1 CPOS (fetch (LOCALCIRCLE RADIUSPOSITION) of LCIRCLE)
                         (fetch (LOCALCIRCLE RADIUS) of LCIRCLE)
                         (fetch (LOCALCIRCLE LOCALCIRCLEBRUSH) of LCIRCLE)
                         DASHING WINDOW])
```

(CLOSED.WIRE.DRAWFN

```
  [LAMBDA (CLOSEDWIREELT WIN REG OPERATION)                      ; Edited 25-May-90 15:26 by matsuda
                                                                 (* draws a closed wire element.)
    (PROG ((GINDVELT (fetch (SCREENELT INDIVIDUALGLOBALPART) of CLOSEDWIREELT))
           (LOCALPART (fetch (SCREENELT LOCALPART) of CLOSEDWIREELT))
           VARX)
          (SETQ VARX (fetch (LOCALCLOSEDWIRE LOCALCLOSEDWIREFILLING) of LOCALPART))
          [COND
             ((OR (fetch (SKFILLING FILLING.TEXTURE) of VARX)
                  (fetch (SKFILLING FILLING.COLOR) of VARX))     (* if there isn't any filling, don't fill.)
              (FILLPOLYGON (fetch (LOCALCLOSEDWIRE KNOTS) of LOCALPART)
                     [COND
                        ((fetch (SKFILLING FILLING.COLOR) of VARX))
                        ((SKETCHINCOLORP)
                         VARX)
                        (T                                       (* simulate color)
```

```
                        (TEXTUREOFCOLOR (fetch (SKFILLING FILLING.COLOR) of VARX]
                    WIN
                    (COND
                       ((EQ (DSPOPERATION NIL WIN)
                            'ERASE)                                  (* if the stream is erasing, erase.)
                        'ERASE)
                       (T                                            (* otherwise use the element's mode.)
                          (fetch (SKFILLING FILLING.OPERATION) of VARX]
          (OR (EQ (fetch (BRUSH BRUSHSIZE) of (SETQ VARX (fetch (LOCALCLOSEDWIRE LOCALCLOSEDWIREBRUSH)
                                                                of LOCALPART)))
                  0)
              (WB.DRAWLINE CLOSEDWIREELT WIN REG OPERATION T (fetch (CLOSEDWIRE CLOSEDWIREDASHING) of GINDVELT)
                     VARX])
```

(SKETCHINCOLORP

```
[LAMBDA NIL                                                          ; Edited 25-May-90 14:00 by matsuda
                                                                     (* hook to determine if sketch should allow color.)
  (AND SKETCHINCOLORFLG (IGREATERP (FETCH SCBITSPERPIXEL OF \CURSORSCREEN)
                                   1])
```

(NEW.READCOLOR1

```
[LAMBDA (MSG ALLOWNONEFLG NOWCOLOR)                                  ; Edited 25-May-90 10:05 by matsuda
  (LET ((INITCOLOR (AND NOWCOLOR (INSURE.RGB.COLOR NOWCOLOR T)))
        COLORINDEX)
       (SETQ COLORINDEX (PAINTW.READBRUSHTEXTURE))
       (COND
          ((NULL COLORINDEX)
           INITCOLOR)
          ((INSURE.RGB.COLOR (ELT (SCREENCOLORMAP NIL)
                                  COLORINDEX)))
          (T INITCOLOR])
```

(SK.FIGUREIMAGE

```
[LAMBDA (SCRITEMS LIMITREGION REGIONOFINTEREST)                      ; Edited 22-May-90 13:23 by matsuda

        (* returns a bitmap which contains the image of the elements on SCRITEMS.
        And a lower left corner.)

  (RESETFORM (CURSOR WAITINGCURSOR)
         (PROG (REGION DSPSTREAM BITMAP LEFT BOTTOM LIMITDIM)
               (COND
                  ((NULL SCRITEMS)
                   (RETURN)))
               [COND
                  ((SCREENELEMENTP SCRITEMS)                         (* single item case.)
                   (SETQ REGION (SK.ITEM.REGION SCRITEMS)))
                  (T (SETQ REGION (SK.ITEM.REGION (CAR SCRITEMS)))
                     [for SCITEM in (CDR SCRITEMS) do (SETQ REGION (SK.UNIONREGIONS REGION (SK.ITEM.REGION
                                                                                             SCITEM]
                                                                     (* order the elements by priority)
                     (SETQ SCRITEMS (REVERSE (SK.SORT.ELTS.BY.PRIORITY SCRITEMS]
                                                                     (* only some of the points are being moved, reduce the region to
          those.)
               (AND REGIONOFINTEREST (SETQ REGION (OR (INTERSECTREGIONS REGION REGIONOFINTEREST)
                                                      REGION)))
               [COND
                  (LIMITREGION

        (* limit the size of the bitmap. This is used by copy insert functions that do not know how big the thing coming in is.)

                                (COND
                                   ((GREATERP (fetch (REGION WIDTH) of REGION)
                                              (SETQ LIMITDIM (fetch (REGION WIDTH) of LIMITREGION)))
                                                                     (* reduce the width picking out the middle of the region)
                                    (replace (REGION LEFT) of REGION with (PLUS (fetch (REGION LEFT) of REGION)
                                                                                (QUOTIENT (DIFFERENCE
                                                                                            LIMITDIM
                                                                                            (fetch (REGION WIDTH)
                                                                                               of REGION))
                                                                                          2)))
                                    (replace (REGION WIDTH) of REGION with LIMITDIM)))
                                (COND
                                   ((GREATERP (fetch (REGION HEIGHT) of REGION)
                                              (SETQ LIMITDIM (fetch (REGION HEIGHT) of LIMITREGION)))
                                                                     (* reduce the height picking out the middle of the region)
                                    (replace (REGION BOTTOM) of REGION with (PLUS (fetch (REGION BOTTOM) of REGION)
                                                                                  (QUOTIENT (DIFFERENCE
                                                                                              LIMITDIM
                                                                                              (fetch (REGION HEIGHT)
                                                                                                 of REGION))
                                                                                            2)))
                                    (replace (REGION HEIGHT) of REGION with LIMITDIM]
                                                                     (* ADD1 is used to convert the possibly floating region
                                                                     coordinates into fixed.)
```

```
                [SETQ DSPSTREAM (DSPCREATE (SETQ BITMAP (BITMAPCREATE (ADD1 (fetch (REGION WIDTH) of REGION))
                                                                      (ADD1 (fetch (REGION HEIGHT) of REGION))
                                                                      (FETCH SCBITSPERPIXEL OF \CURSORSCREEN]
                (DSPXOFFSET [IMINUS (SETQ LEFT (FIXR (fetch (REGION LEFT) of REGION]
                       DSPSTREAM)
                (DSPYOFFSET [IMINUS (SETQ BOTTOM (FIXR (fetch (REGION BOTTOM) of REGION]
                       DSPSTREAM)
```

(* this is because the default clipping region is smaller than the clipping region of the figure in extreme cases.)

```
                (DSPCLIPPINGREGION REGION DSPSTREAM)
                (DSPOPERATION 'PAINT DSPSTREAM)                (* to avoid carriage returns.)
                (DSPRIGHTMARGIN (PLUS 100 (fetch (REGION RIGHT) of REGION))
                       DSPSTREAM)
                (DRAW.LOCAL.SKETCH SCRITEMS DSPSTREAM REGION)
                (RETURN (create SKFIGUREIMAGE
                               SKFIGURE.LOWERLEFT _ (create POSITION
                                                            XCOORD _ LEFT
                                                            YCOORD _ BOTTOM)
                               SKFIGURE.BITMAP _ BITMAP])

)

(MOVD 'READCOLOR1 'ORG.READCOLOR1)

(MOVD 'NEW.READCOLOR1 'READCOLOR1)

(REPLACE (IMAGEOPS IMFILLPOLYGON) OF \8DISPLAYIMAGEOPS WITH (FUNCTION POLYSHADE.DISPLAY))

(REPLACE (IMAGEOPS IMSCALEDBITBLT) OF \8DISPLAYIMAGEOPS WITH (FUNCTION \SCALEDBITBLT.DISPLAY))

(PUTPROPS NEW-SKETCH-COLOR COPYRIGHT ("Fuji Xerox Co., Ltd" 1990))
```

FUNCTION INDEX