

File created: 5-Dec-2023 00:11:01 {WMEDLEY}<library>sketch>SKETCH-BMELT.;1

edit by: rmk

previous date: 24-Mar-92 14:07:17 {WMEDLEY}<library>sketch>SKETCHBMELT.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

(RPAQQ SKETCH-BMELTCOMS

```
(( * BITMAP element done to allow zooming of bitmaps that is not possible with image object bitmaps.)
(FNS SKETCH.CREATE.BITMAP INIT.BITMAP.ELEMENT BITMAPELT.CHANGEFN BITMAPELT.DRAWFN DSPVIEWPORT
SK.COMPUTE.LOCAL.SCALED.BITMAP BITMAPELT.EXPANDFN BITMAPELT.INSIDEFN BITMAPELT.TRANSLATEFN
BITMAPELT.REGIONFN BITMAPELT.GLOBALREGIONFN BITMAPELT.READCHANGEFN BITMAPELT.TRANSFORMFN
SK.BITMAP.CREATE.BITMAP.SET.SCALES BITMAPELT.INPUTFN BITMAPELT.CHOOSE.BITMAP)
(DECLARE%: DONTCOPY (RECORDS BITMAPELT LOCALBITMAPELT))
(FILESCALEBITMAP))
```

(* * BITMAP element done to allow zooming of bitmaps that is not possible with image object bitmaps.)

(DEFINEQ

(SKETCH.CREATE.BITMAP

```
[LAMBDA (BITMAP POSITION SCALE SCALECACHE PRIORITY) (* rrb "13-Mar-86 17:30")
(* creates a sketch bitmap element.)
(SK.BITMAP.CREATE (OR (BITMAPP BITMAP)
(\ILLEGAL.ARG BITMAP))
(SK.INSURE.POSITION POSITION)
(OR (NUMBERP SCALE)
1.0)
[COND
(SCALECACHE (for CACHE in SCALECACHE do (OR (AND (NUMBERP (CAR CACHE))
(BITMAPP (CADR CACHE)))
(\ILLEGAL.ARG CACHE)))
(SORT SCALECACHE (FUNCTION (LAMBDA (A B)
(GREATERP A B)
PRIORITY])])
```

(INIT.BITMAP.ELEMENT

```
[LAMBDA NIL (* rrb "18-Oct-85 17:17")
(* creates a bitmap element. This will scale bitmaps)
(COND
((NOT (SKETCH.ELEMENT.TYEP 'BITMAPELT))
(CREATE.SKETCH.ELEMENT.TYEP 'BITMAPELT "Bit image" "prompts for a region of the screen as a scalable
bitmap." (FUNCTION BITMAPELT.DRAWFN)
(FUNCTION BITMAPELT.EXPANDFN)
'OBSOLETE
(FUNCTION BITMAPELT.CHANGEFN)
(FUNCTION BITMAPELT.INPUTFN)
(FUNCTION BITMAPELT.INSIDEFN)
(FUNCTION BITMAPELT.REGIONFN)
(FUNCTION BITMAPELT.TRANSLATEFN)
NIL
(FUNCTION BITMAPELT.READCHANGEFN)
(FUNCTION BITMAPELT.TRANSFORMFN)
NIL
(FUNCTION BITMAPELT.GLOBALREGIONFN])
```

(BITMAPELT.CHANGEFN

```
[LAMBDA (SCRELTS SKW HOW) (* rrb "11-Jul-86 15:51")
(* changefn for scaleable bitmaps.
Only works on the first bitmap for now.)
(PROG ((BMELT (AND (EQ (fetch (SCREENELT GTYPE) of (CAR SCRELTS))
'BITMAPELT)
(CAR SCRELTS)))
GBMELT INDGBMELT NEWBM BMCACHEENTRY BM ORIGBM BMREGION BMSCALE ORIGSCALE BMCACHE NEWSCALE NEWVALUE
ELTPRI)
(OR BMELT (RETURN))
(SETQ INDGBMELT (fetch (SCREENELT INDIVIDUALGLOBALPART) of BMELT))
[SETQ ELTPRI (SK.ELEMENT.PRIORITY (SETQ GBMELT (fetch (SCREENELT GLOBALPART) of BMELT)
(SETQ ORIGBM (fetch (BITMAPELT SKBITMAP) of INDGBMELT))
(SETQ ORIGSCALE (fetch (BITMAPELT SKBITMAPSCALE) of INDGBMELT))
(* get the bitmap that generated the image the user was seeing.)
(SETQ BMCACHEENTRY (fetch (LOCALBITMAPELT SOURCEFORIMAGE) of (fetch (SCREENELT LOCALPART) of BMELT)))
(SETQ BMSCALE (CAR BMCACHEENTRY))
(SETQ BM (CADR BMCACHEENTRY))
(SETQ BMREGION (fetch (BITMAPELT SKBITMAPREGION) of INDGBMELT))
(SETQ BMCACHE (fetch (BITMAPELT SKBITMAPPCACHE) of INDGBMELT))
(RETURN (AND (SETQ NEWBM
(SELECTQ HOW
(EDIT (* call the bitmap editor and if changes are made, recreate the
element)
```

```
(AND (SETQ NEWBM (EDIT.BITMAP BM))
      (create SKHISTORYCHANGESPEC
              NEWELT _ (COND
                        ((EQ BM ORIGBM)
                         (SK.BITMAP.CREATE NEWBM (create POSITION
                                                         XCOORD _
                                                         (fetch (REGION LEFT)
                                                             of BMREGION)
                                                         YCOORD _
                                                         (fetch (REGION BOTTOM)
                                                             of BMREGION))
                         (ORIGSCALE BMCACHE ELTPRI)))
                        (T (* clobber the cache and redo the image.)
                          (RPLACA (CDR BMCACHEENTRY)
                                   NEWBM)
                          (SK.BITMAP.CREATE ORIGBM
                                              (create POSITION
                                                      XCOORD _ (fetch (REGION LEFT)
                                                                      of BMREGION)
                                                      YCOORD _ (fetch (REGION BOTTOM)
                                                                      of BMREGION))
                                              (ORIGSCALE BMCACHE ELTPRI)))
                        (OLDELT _ GBMELT
                          PROPERTY _ 'DATA
                          NEWVALUE _ NEWBM
                          OLDVALUE _ ORIGBM)))
      (CHANGEBITMAP
        (COND
          ((EQ ORIGBM BM)
           (create SKHISTORYCHANGESPEC
                   NEWELT _ (SK.BITMAP.CREATE (fetch (LOCALBITMAPELT LOCALBITMAP)
                                                       of (fetch (SCREENELT LOCALPART)
                                                             of BMELT)))
                   (create POSITION
                           XCOORD _ (fetch (REGION LEFT)
                                             of BMREGION)
                           YCOORD _ (fetch (REGION BOTTOM)
                                             of BMREGION))
                   (VIEWER.SCALE SKW)
                   BMCACHE ELTPRI)
           (OLDELT _ GBMELT
             PROPERTY _ 'SCALE
             NEWVALUE _ (VIEWER.SCALE SKW)
             OLDVALUE _ (ORIGSCALE)))
          (T (* clobber cache and redraw)
            (RPLACA BMCACHEENTRY (VIEWER.SCALE SKW))
            (create SKHISTORYCHANGESPEC
                    NEWELT _ (SK.BITMAP.CREATE ORIGBM
                                                (create POSITION
                                                        XCOORD _ (fetch (REGION LEFT)
                                                                    of BMREGION)
                                                        YCOORD _ (fetch (REGION BOTTOM)
                                                                    of BMREGION))
                                                (ORIGSCALE BMCACHE ELTPRI))
                    (OLDELT _ GBMELT
                      PROPERTY _ 'CACHE
                      NEWVALUE _ BMCACHE
                      OLDVALUE _ BMCACHE))))
      (CHANGEBITMAP&SCALE
```

(* makes the image shown be the original bitmap at the original scale.
Provides a way of expanding the bitmap.)

(* rather than figure out what the cache should do here just flush it.
Maybe should be scaled but too lazy now.)

```
(create SKHISTORYCHANGESPEC
        NEWELT _ (SK.BITMAP.CREATE (fetch (LOCALBITMAPELT LOCALBITMAP)
                                          of (fetch (SCREENELT LOCALPART)
                                                of BMELT)))
        (create POSITION
                XCOORD _ (fetch (REGION LEFT) of BMREGION)
                YCOORD _ (fetch (REGION BOTTOM) of BMREGION))
        BMSCALE NIL ELTPRI)
(OLDELT _ GBMELT
  PROPERTY _ 'DATA
  NEWVALUE _ (fetch (LOCALBITMAPELT LOCALBITMAP) of (fetch (SCREENELT LOCALPART)
                                                            of BMELT))
  OLDVALUE _ ORIGBM))
(CHANGESCALE (* make the bitmap have this as its current scale.)
```

(* rather than figure out what the cache should do here just flush it.
Maybe should be scaled but too lazy now.)

```

(create SKHISTORYCHANGESPEC
  NEWELT _ (SK.BITMAP.CREATE ORIGBM (create POSITION
    XCOORD _
    (fetch (REGION LEFT)
      of BMREGION)
    YCOORD _
    (fetch (REGION BOTTOM)
      of BMREGION))
    (VIEWER.SCALE SKW)
    BMCACHE NIL ELTPRI)
  OLDELТ _ GBMELT
  PROPERTY _ 'SCALE
  NEWVALUE _ (VIEWER.SCALE SKW)
  OLDVALUE _ ORIGSCALE))
(CACHE (COND
  ((AND (NOT (EQP (SETQ NEWSCALE (VIEWER.SCALE SKW))
    ORIGSCALE))
    (NOT (SASSOC NEWSCALE BMCACHE))))
    (* make sure there isn't already a cache at this scale.)
  (create SKHISTORYCHANGESPEC
    NEWELT _ (SK.BITMAP.CREATE
      ORIGBM
      (create POSITION
        XCOORD _ (fetch (REGION LEFT) of BMREGION)
        YCOORD _ (fetch (REGION BOTTOM) of BMREGION))
      ORIGSCALE
      [SETQ NEWVALUE
        (SORT (CONS (LIST NEWSCALE (fetch (LOCALBITMAPELT
          LOCALBITMAP)
            of (fetch (SCREENELT
              LOCALPART
                )
              of BMELT))))
          (APPEND BMCACHE))
        (FUNCTION (LAMBDA (A B)
          (GREATERP (CAR A)
            (CAR B)
              ELTPRI)
            OLDELТ _ GBMELT
            PROPERTY _ 'CACHE
            NEWVALUE _ NEWVALUE
            OLDVALUE _ BMCACHE))))))
  (DELETECACHE (COND
    ((EQ BM ORIGBM) (* wants to delete the original, replace it with a nearby cache.)
      (STATUSPRINT SKW "Not implemented to delete the original. If you
        really want to, you can change the original with the other
        bitmap change edit commands.)))
    (T (create SKHISTORYCHANGESPEC
      NEWELT _ (SK.BITMAP.CREATE ORIGBM
        (create POSITION
          XCOORD _ (fetch (REGION LEFT)
            of BMREGION)
          YCOORD _ (fetch (REGION BOTTOM)
            of BMREGION))
          ORIGSCALE
          (SETQ NEWVALUE (REMOVE BMCACHEENTRY
            BMCACHE))
          ELTPRI)
        OLDELТ _ GBMELT
        PROPERTY _ 'CACHE
        NEWVALUE _ NEWVALUE
        OLDVALUE _ BMCACHE))))))
  NIL))
(LIST NEWBM])

```

(BITMAPELT.DRAWFN

[LAMBDA (BITMAPELT WINDOW

; Edited 24-Mar-92 13:59 by jds

;; shows a bitmap element. The local bitmap is only computed and cached for streams that don't support a scaled bitblt operation.

```

(PROG ((GLOBALBMELT (fetch (SCREENELT INDIVIDUALGLOBALPART) of BITMAPELT))
  (LOCALBMELT (fetch (SCREENELT LOCALPART) of BITMAPELT))
  BITMAP)
  (RETURN (COND
    [(OR (IMAGESTREAMTYPEPEP WINDOW 'INTERPRESS)
      (SETQ BITMAP (fetch (LOCALBITMAPELT LOCALBITMAP) of LOCALBMELT)))]
      ; INTERPRESS has a SCALED BITBLT operation but it doesn't
      ; work so don't use it.
    (PROG (LOCALREGION VISIBLEREGION IMAGEREGION)
      ;; make sure the local region of the current cached image completely covers the visible part of the bitmap. This allows us
      ;; to only compute the visible portion of large bitmaps.
      (SETQ LOCALREGION (fetch (LOCALBITMAPELT LOCALBITMAPREGION) of LOCALBMELT))
      ; if nothing is visible, don't do anything. This may happen if the
      ; bitmap is part of a group.
      (OR (SETQ VISIBLEREGION (INTERSECTREGIONS (fetch (LOCALBITMAPELT LOCALBITMAPREGION)

```



```

SCALEAMOUNT)))
(SETQ LBOT (FIX (QUOTIENT (DIFFERENCE (fetch (REGION BOTTOM) of LOCALPIECE)
(fetch (REGION BOTTOM) of LOCALREGION))
SCALEAMOUNT))) (* copy the piece of bitmap that we want into an auxiliary to be scaled.)
[SETQ SBM (BITMAPCREATE (FIX (PLUS (QUOTIENT (fetch (REGION WIDTH) of LOCALPIECE)
SCALEAMOUNT)
1.0))
(FIX (PLUS (QUOTIENT (fetch (REGION HEIGHT) of LOCALPIECE)
SCALEAMOUNT)
1.0)]
(BITBLT (CADR BMCACHE)
LLEFT LBOT SBM)
(RETURN (LIST (CREATEREGION (PLUS (fetch (REGION LEFT) of LOCALREGION)
(QUOTIENT LLEFT LOCALSCALE))
(PLUS (fetch (REGION BOTTOM) of LOCALREGION)
(QUOTIENT LBOT LOCALSCALE))
(BITMAPWIDTH SBM)
(BITMAPHEIGHT SBM))
(SCALEBITMAP SBM SCALEAMOUNT]))

```

(BITMAPELT.EXPANDFN

```

[LAMBDA (GBITMAPELT SCALE STREAM) (* rrb "11-Jul-86 15:55")
(* creates a local bitmap screen element from a global bitmap element.)
(PROG ((INDGBMELT (fetch (GLOBALPART INDIVIDUALGLOBALPART) of GBITMAPELT))
LOCALBITMAPREGION BMSCALE BMCACHE)
[SETQ BMSCALE (QUOTIENT (fetch (BITMAPELT SKBITMAPSCALE) of INDGBMELT)
(TIMES SCALE (DSPSCALE NIL STREAM)
(SETQ LOCALBITMAPREGION (SK.SCALE.REGION (fetch (BITMAPELT SKBITMAPREGION) of INDGBMELT)
SCALE))
(SETQ BMCACHE (BITMAPELT.CHOSE.BITMAP (fetch (GLOBALPART INDIVIDUALGLOBALPART) of GBITMAPELT)
SCALE))
(RETURN (create SCREENELT
LOCALPART _ (create LOCALBITMAPELT
LOCALBITMAPPOSITION _ (create POSITION
XCOORD _ (fetch (REGION LEFT)
of LOCALBITMAPREGION)
YCOORD _ (fetch (REGION BOTTOM)
of LOCALBITMAPREGION))
LOCALBITMAP _ (COND
((OR (MEMB (fetch (IMAGEOPS IMSCALEDBITBLT)
of (fetch (STREAM IMAGEOPS)
of STREAM))
' (NIL NIL))
(IMAGESTREAMTYPEP STREAM 'PRESS))

```

(* see if the stream supports scaled bitblt This assumes that windows don't and will have to be changed when they do. Spruce printers don't implement scaled bitblt even though the image ops vector has an entry that works for full press. Since diagonal lines and curves don't work to full press, let's just make everything work as best possible to Spruce. Also the scale bitblt operation for Interpress doesn't work; there is code in BITMAPELT.DRAWFN to hack around this.)
(* the actual bitmap to be displayed will be computed by the display fn.)

```

T)
(T (* if stream implements scaled bitblt, not need for caching a scaled bitmap.)
NIL)
LOCALBITMAPREGION _ LOCALBITMAPREGION
LOCALSCALE _ SCALE
SOURCEFORIMAGE _ BMCACHE)
GLOBALPART _ GBITMAPELT])

```

(BITMAPELT.INSIDFN

```

[LAMBDA (GBMELT WREG) (* rrb "28-Sep-85 19:43")
(* determines if the global bitmap element GBMELT is inside of
WREG.)
(REGIONSINTERSECTP (fetch (BITMAPELT SKBITMAPREGION) of (fetch (GLOBALPART INDIVIDUALGLOBALPART) of GBMELT))
WREG])

```

(BITMAPELT.TRANSLATEFN

```

[LAMBDA (SKELT DELTAPOS) (* rrb "28-Sep-85 19:49")
(* * returns a bitmap element which has the bitmap translated by DELTAPOS)
(PROG ((GBMELT (fetch (GLOBALPART INDIVIDUALGLOBALPART) of SKELT))
(RETURN (create GLOBALPART
COMMONGLOBALPART _ (APPEND (fetch (GLOBALPART COMMONGLOBALPART) of SKELT))
INDIVIDUALGLOBALPART _ (create BITMAPELT using
GBMELT SKBITMAPREGION _
(REL.MOVE.REGION (fetch (BITMAPELT
SKBITMAPREGION)
of GBMELT)
(fetch (POSITION XCOORD)
of DELTAPOS)

```

(fetch (POSITION YCOORD) of DELTAPOS])

(BITMAPELT.REGIONFN

[LAMBDA (BMSCRLET)

(* rrb "28-Sep-85 19:45")

(* returns the region occupied by a bitmap)

(fetch (LOCALBITMAPELT LOCALBITMAPREGION) of (fetch (SCREENELT LOCALPART) of BMSCRLET])

(BITMAPELT.GLOBALREGIONFN

[LAMBDA (GBITMAPELT)

(* rrb "18-Oct-85 17:17")

(* returns the global region occupied by a global bitmap element.)

(fetch (BITMAPELT SKBITMAPREGION) of (fetch (GLOBALPART INDIVIDUALGLOBALPART) of GBITMAPELT])

(BITMAPELT.READCHANGEFN

[LAMBDA (SKW SCRNELTS)

(* rrb "11-Jul-86 15:51")

(* the users has selected SCRNELT to be changed this function reads a specification of how the bitmap elements should change.)

(* if the bitmap is at its original scale,let the user edit it like an image object bitmap. If it isn't, give them the option of moving it to this scale, making the one shown be the original one or EDIT which informs them they must do one of the other two.)

(PROG [(BMELT (for ELT in SCRNELTS when (EQ (fetch (SCREENELT GTYPE) of ELT) 'BITMAPELT)

do (RETURN (fetch (SCREENELT INDIVIDUALGLOBALPART) of ELT])

(RETURN (COND

((EQUAL (fetch (BITMAPELT SKBITMAPSCALE) of BMELT)

(VIEWER.SCALE SKW))

(* do bitmap editor.)

'EDIT)

(T (\CURSOR.IN.MIDDLE.MENU (create MENU

CENTERFLG _ T

TITLE _ "Scaled bitmap operations"

ITEMS _ (APPEND ' ("Perform edit operations on the source bitmap of this image." 'EDIT

"Allows editing of the original or cached bitmap. Result will be scaled back into this image.")

("Make the image shown be the source" 'CHANGEBITMAP "Replaces the original or cached bitmap that is the source of this image with this image at this scale. Further scaling are done from this image.")

("Make the source be at this scale" 'CHANGESCALE "changed the scale of the original or cached bitmap to be at this scale.")

("Make the image shown be the source at the source scale" 'CHANGEBITMAP&SCALE "makes it as if the bitmap image shown had been input at the original scale.")

("Save this image to be used as a source at this scale" 'CACHE "This image will be saved and used when displaying the image at this scale.

It can then be edited without effecting the original.")

(AND (fetch (BITMAPELT SKBITMAPCACHE) of BMELT)

' ("Remove this source from the cache." 'DELETECACHE "Removes the source of this image from the cache. The image will then come from the nearest other source."])

(BITMAPELT.TRANSFORMFN

[LAMBDA (GELT TRANSFORMFN TRANSFORMDATA SCALEFACTOR)

(* rrb "13-Mar-86 17:31")

(* returns a copy of the global bitmap element that has its control point transformed by transformfn. TRANSFORMDATA is arbitrary data that is passed to tranformfn.)

(PROG ((INDVPART (fetch (GLOBALPART INDIVIDUALGLOBALPART) of GELT))

GREG)

(RETURN (SK.BITMAP.CREATE (fetch (BITMAPELT SKBITMAP) of INDVPART)

(SK.TRANSFORM.POINT (create POSITION

XCOORD _ (fetch (REGION LEFT) of (SETQ GREG

(fetch (BITMAPELT

```

                                SKBITMAPREGION)
                                of INDVPART)))
                                YCOORD _ (fetch (REGION BOTTOM) of GREG))
TRANSFORMFN TRANSFORMDATA)
(TIMES (fetch (BITMAPELT SKBITMAPSCALE) of INDVPART)
SCALEFACTOR])

```

(SK.BITMAP.CREATE

```

[LAMBDA (BITMAP POSITION INITSCALE CACHELST PRIORITY)
(* rrb "13-Mar-86 17:29")
(* creates a BITMAPELT sketch element)

(PROG (NEWBMELT)
(SETQ NEWBMELT (create GLOBALPART
INDIVIDUALGLOBALPART _ (create BITMAPELT
SKBITMAP _ BITMAP
SKBITMAPREGION _ (CREATEREGION (fetch (POSITION
XCOORD)
of POSITION)
(fetch (POSITION YCOORD)
of POSITION)
(TIMES (BITMAPWIDTH BITMAP)
)
INITSCALE)
(TIMES (BITMAPHEIGHT
BITMAP)
INITSCALE)))

SKBITMAPSCALE _ INITSCALE
SKBITMAPCACHE _ CACHELST)))

(BITMAP.SET.SCALES NEWBMELT)
(AND PRIORITY (SK.SET.ELEMENT.PRIORITY NEWBMELT))
(RETURN NEWBMELT])

```

(BITMAP.SET.SCALES

```

[LAMBDA (GBMELT)
(* rrb "17-Oct-85 17:34")
(* updates the scale field after a change in a bitmap element.)

(PROG ((GREG (fetch (BITMAPELT SKBITMAPREGION) of (fetch (GLOBALPART INDIVIDUALGLOBALPART) of GBMELT)))
WIDTH HEIGHT)
(replace (GLOBALPART MINSCALE) of GBMELT with (FQUOTIENT (MIN (SETQ WIDTH (fetch (REGION WIDTH)
of GREG))
(SETQ HEIGHT (fetch (REGION HEIGHT)
of GREG)))
1000.0))
(replace (GLOBALPART MAXSCALE) of GBMELT with (FQUOTIENT (MAX WIDTH HEIGHT)
2.0))

(RETURN GBMELT])

```

(BITMAPELT.INPUTFN

```

[LAMBDA (WINDOW)
(* rrb "11-Jul-86 15:51")
(* 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)))
(BITBLT (SCREENBITMAP)
(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]))

```

(BITMAPELT.CHOOSE.BITMAP

```

[LAMBDA (GBMELT SCALE)
(* rrb "17-Oct-85 17:50")
(* chooses the closest bitmap image from the cache and returns

a list of (itsscale bitmap))
(PROG ((CACHE (fetch (BITMAPELT SKBITMAPCACHE) of GBMELT))
(ORIGSCALE (fetch (BITMAPELT SKBITMAPSCALE) of GBMELT))
(ORIGBITMAP (fetch (BITMAPELT SKBITMAP) of GBMELT))
GREATER LESSER)
[COND
((OR (NULL CACHE)
(EQP ORIGSCALE SCALE))
(RETURN (LIST ORIGSCALE ORIGBITMAP)
(* special case)
(* find the bounding cached values)
[for PR in CACHE do (COND
((GREATERP (CAR PR)
SCALE)
(SETQ GREATER PR))
(T (SETQ LESSER PR)
(RETURN)
[COND

```

[(GREATERP ORIGSCALE SCALE)

(* the original is larger than this scale, see how it compares to the greater one found in the cache.)

(COND
 [GREATER (COND
 ((EQP (CAR LESSER)
 SCALE) (* special check since LESSER might have been equal.)
 (RETURN LESSER))
 ((LESSP ORIGSCALE (CAR GREATER))
 (SETQ GREATER (LIST ORIGSCALE ORIGBITMAP))
 (T (SETQ GREATER (LIST ORIGSCALE ORIGBITMAP))
 [LESSER

(* the original is smaller than this scale, see how it compares to the lesser one found in the cache.)

(COND
 ((GREATERP ORIGSCALE (CAR LESSER))
 (SETQ LESSER (LIST ORIGSCALE ORIGBITMAP)))
 ((EQP (CAR LESSER)
 SCALE) (* special check since LESSER might have been equal.)
 (RETURN LESSER]
 (T (SETQ LESSER (LIST ORIGSCALE ORIGBITMAP)) (* GREATER is scaled just greater than SCALE.
 LESSER is just less. Choose between them.)
 (RETURN (COND
 (GREATER (COND
 (LESSER (* pick closest one)
 (COND
 ((GREATERP SCALE (QUOTIENT (PLUS (CAR LESSER)
 (CAR GREATER))
 2))
 GREATER)
 (T LESSER)))
 (T GREATER)))
 (T LESSER]))

)

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE

(TYPERECORD BITMAPELT (SKBITMAP SKBITMAPREGION SKBITMAPSCALE SKBITMAPCACHE))

(RECORD LOCALBITMAPELT ((LOCALBITMAPPOSITION)
 LOCALHOTREGION LOCALBITMAP LOCALBITMAPREGION (* coordinates of entire region covered by the bitmap in local
 coordinates.)
 LOCALSCALE SOURCEFORIMAGE (* pair of the scale and cached image from which
 LOCALBITMAP was generated.)
 LOCALIMAGEREGION

(* region in local coordinates of the area covered by LOCALBITMAP.
 This may be a subregion of LOCALBITMAPREGION)

))

)

)

(FILESLOAD SCALEBITMAP)

FUNCTION INDEX

BITMAP.SET.SCALES	7	BITMAPELT.INPUTFN	7	DSPVIEWPORT	4
BITMAPELT.CHANGEFN	1	BITMAPELT.INSIDFN	5	INIT.BITMAP.ELEMENT	1
BITMAPELT.CHOOSE.BITMAP	7	BITMAPELT.READCHANGEFN	6	SK.BITMAP.CREATE	7
BITMAPELT.DRAWFN	3	BITMAPELT.REGIONFN	6	SK.COMPUTE.LOCAL.SCALED.BITMAP ..	4
BITMAPELT.EXPANDFN	5	BITMAPELT.TRANSFORMFN	6	SKETCH.CREATE.BITMAP	1
BITMAPELT.GLOBALREGIONFN	6	BITMAPELT.TRANSLATEFN	5		

RECORD INDEX

BITMAPELT	8	LOCALBITMAPELT	8
-----------------	---	----------------------	---
