*File created:* 26-Jan-88 17:58:05 **{QV}<TRIGG>LISP>STYLESHEET.;2**

*changes to:* (FNS STYLESHEET.CHANGE.ITEMS STYLESHEET.FILL.IN.WINDOW STYLESHEET.IMAGEHEIGHT STYLESHEET.IMAGEWIDTH STYLESHEET.AT.LOAD)
(VARS STYLESHEETCOMS)

*previous date:* 27-Aug-87 13:10:53 {IE:PARC:XEROX}<LISPUSERS>LYRIC>STYLESHEET.;1

*Read Table:* INTERLISP

*Package:* INTERLISP

*Format:* XCCS

;;


```
(RPAQQ STYLESHEETCOMS
       [(DECLARE%: DONTCOPY (PROPS (STYLESHEET MAKEFILE-ENVIRONMENT)
                                   (STYLESHEET FILETYPE)))
        (DECLARE%: (LOCALVARS . T))
        (* * Public entry)
        (FNS CREATE.STYLE STYLESHEETP STYLE.PROP STYLESHEET STYLESHEET.IMAGEHEIGHT STYLESHEET.IMAGEWIDTH)
        (* * Private routines.)
        (DECLARE%: EVAL@COMPILE DONTCOPY (RECORDS STYLEBLOCK))
        (INITRECORDS STYLEBLOCK)
        (FNS STYLESHEET.CHANGE.FILL STYLESHEET.CHANGE.ITEMS STYLESHEET.CHANGE.SELECTIONS
             STYLESHEET.CHANGE.TITLES STYLESHEET.MENUITEM)
        (FNS STYLESHEET.SETUP STYLESHEET.WAIT.TILL.DONE STYLESHEET.GET.SELECTIONS STYLESHEET.CLEANUP)
        (FNS STYLESHEET.WHENSELECTEDFN STYLESHEET.CLEAR.WHENSELECTEDFN STYLESHEET.DONE.FN)
        (FNS STYLESHEET.ITEM.HEIGHT STYLESHEET.ITEM.WIDTH)
        (FNS STYLESHEET.CREATE.WINDOW STYLESHEET.FILL.IN.WINDOW STYLESHEET.ADD.MENU STYLESHEET.SHADE.SELECTIONS)
        (FNS STYLESHEET.AT.LOAD)
        (P (STYLESHEET.AT.LOAD))
        (GLOBALVARS STYLESHEET.SELECTED.SHADE)
        (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                                (NLAML)
                                                                                (LAMA STYLE.PROP CREATE.STYLE])

(DECLARE%: DONTCOPY

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

(PUTPROPS STYLESHEET FILETYPE :TCOMPL)
)

(DECLARE%:

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)
)
```

(* * Public entry)

```
(DEFINEQ
```

## (CREATE.STYLE

```
  [LAMBDA STYLE                                                    (* hts%: "21-Mar-85 14:14")
```

(* * Stylesheet constructor.)

```
    (LET* [(PLIST (for I to STYLE collect (ARG STYLE I)))
           (ITEMS (LISTP (LISTGET PLIST 'ITEMS]
          (if ITEMS
              then (LET ((STYLESHEET (LIST 'STYLESHEET 'ITEMS ITEMS)))
                        (for PROP on PLIST by (CDDR PROP) do (STYLE.PROP STYLESHEET (CAR PROP)
                                                                         (CADR PROP)))
                        STYLESHEET)
            else NIL])
```

## (STYLESHEETP

```
  [LAMBDA (THING)                                                  (* hts%: "22-Mar-85 13:00")
```

(* * Tells whether THING is probably a stylesheet.)

```
    (AND (LISTP THING)
         (EQ (CAR THING)
             'STYLESHEET)
         (LISTP (LISTGET (CDR THING)
                         'ITEMS])
```

## (STYLE.PROP

```
  [LAMBDA PROPSTUFF                                                    (* hts%: "22-Mar-85 13:22")
```

(* * Get or put a "field" of a stylesheet. Works externally the same way as WINDOWPROP.
A stylesheet is represented as the cons of the atom STYLESHEET and a plist containing all the requisite data.)

```
    (OR (EQ PROPSTUFF 2)
        (EQ PROPSTUFF 3)
        (\ILLEGAL.ARG))
    (LET ((STYLESHEET (ARG PROPSTUFF 1)))
         (OR (STYLESHEETP STYLESHEET)
             (\ILLEGAL.ARG STYLESHEET))
         (LET [(PROP (MKATOM (ARG PROPSTUFF 2]
              (if (EQ PROPSTUFF 2)
                  then (LISTGET (CDR STYLESHEET)
                                PROP)
                else (LET ((OLDVAL (LISTGET (CDR STYLESHEET)
                                            PROP))
                           (NEWVAL (ARG PROPSTUFF 3)))
                          (LISTPUT (CDR STYLESHEET)
                                   PROP NEWVAL)
```

(* * Certain stylesheet properties are normalized and collected into a list of "styleblocks" %, one for each item.
Styleblocks make it easier later on to handle the information for each item.
Styleblocks store the items themselves, selections, fill information, titles, and CLEAR-ALL submenus
(if any)%.)

```
                          (SELECTQ (ARG PROPSTUFF 2)
                               (ITEMS (STYLESHEET.CHANGE.ITEMS STYLESHEET NEWVAL))
                               (NEED.NOT.FILL.IN
                                    (STYLESHEET.CHANGE.FILL STYLESHEET NEWVAL))
                               (SELECTIONS (STYLESHEET.CHANGE.SELECTIONS STYLESHEET NEWVAL))
                               (ITEM.TITLES (STYLESHEET.CHANGE.TITLES STYLESHEET NEWVAL))
                               NIL)
                          OLDVAL])
```

## (STYLESHEET

```
  [LAMBDA (STYLE)                                                      ; Edited 27-Aug-87 13:08 by Stansbury
```

;;; Creates a window, lays out the menus in it, and waits for BACKGROUND to notify it that the the user has made all his selections and hit the DONE
;;; button. Then removes the window and returns the selections the user made.

```
    (OR (STYLESHEETP STYLE)
        (\ILLEGAL.ARG STYLE))
    (LET ((OLD.WHENSELECTEDFNS (STYLESHEET.SETUP STYLE)          ; Hold onto old WHENSELECTEDFNs so that they can be
                                                                 ; restored on exit.

          )
         (W (STYLESHEET.CREATE.WINDOW STYLE)                     ; Lay out stylesheet of appropriate size and fill it in.

         ))
        ;; Wait until the user has filled everything in he needs to, and has hit the DONE button.
        (STYLESHEET.WAIT.TILL.DONE W STYLE)
        ;; Clean things up and return user's selections.
        (PROG1 (if (EQ (WINDOWPROP W 'HOW NIL)
                       'ABORT)
                   then ;; user selected ABORT: restore original selections, in case stylesheet is reused. Return NIL.
                        (STYLE.PROP STYLE 'SELECTIONS (STYLE.PROP STYLE 'SELECTIONS))
                        NIL
                 else ;; normal exit: return new selections.
                      (STYLESHEET.GET.SELECTIONS STYLE))
               (STYLESHEET.CLEANUP W STYLE OLD.WHENSELECTEDFNS])
```

## (STYLESHEET.IMAGEHEIGHT

```
  [LAMBDA (STYLESHEET)                                                 ; Edited 26-Jan-88 11:48 by Trigg
```

;;; Tells how high in pixels the given stylesheet would be if it were displayed.

;; rht 1/26/88: Now grabs Done/Reset/Abort styleblock off of STYLEPROP rather than from globalvar.

```
    (HEIGHTIFWINDOW (bind THIS.ONE (BIG _ 0)
                          (TITLEFONT _ (STYLE.PROP STYLESHEET 'ITEM.TITLE.FONT)) for BLOCK
                       in (CONS (STYLE.PROP STYLESHEET '\DONE.STYLEBLOCK)
                                (STYLE.PROP STYLESHEET '\STYLE.BLOCKS))
                       do (if (IGREATERP (SETQ THIS.ONE (STYLESHEET.ITEM.HEIGHT BLOCK TITLEFONT))
                                         BIG)
                              then (SETQ BIG THIS.ONE))
                       finally (RETURN BIG))
          (STYLE.PROP STYLESHEET 'TITLE])
```

## (STYLESHEET.IMAGEWIDTH

```
  [LAMBDA (STYLESHEET)                                    ; Edited 26-Jan-88 11:50 by Trigg

;;; returns the width in pixels this entire stylesheet would take up on the screen were it displayed.
      ;; rht 1/26/88: Now grabs Done/Reset/Abort styleblock off of STYLEPROP rather than from globalvar.
    (WIDTHIFWINDOW (LET [(BLOCKS (STYLE.PROP STYLESHEET '\STYLE.BLOCKS]
                        (IPLUS (bind (TITLEFONT _ (STYLE.PROP STYLESHEET 'ITEM.TITLE.FONT)) for BLOCK
                                  in (CONS (STYLE.PROP STYLESHEET '\DONE.STYLEBLOCK)
                                           BLOCKS)
                                  sum (STYLESHEET.ITEM.WIDTH BLOCK TITLEFONT))
                               (ITIMES 2 (LENGTH BLOCKS])

)


           (* * Private routines.)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(DATATYPE STYLEBLOCK (TITLE MENU SUBMENU FILL SELECTIONS))
)

(/DECLAREDATATYPE 'STYLEBLOCK '(POINTER POINTER POINTER POINTER POINTER)
       ;; ---field descriptor list elided by lister---
       '10)
)

(/DECLAREDATATYPE 'STYLEBLOCK '(POINTER POINTER POINTER POINTER POINTER)
       ;; ---field descriptor list elided by lister---
       '10)

(DEFINEQ

(STYLESHEET.CHANGE.FILL
  [LAMBDA (STYLESHEET NEWFILL BLOCKS)                     (* hts%: "23-Mar-85 16:22")

           (* * Modifies the given stylesheet to reflect new menu fill information.
           Must be called either when the fill info changes or when the styleblock list must be rebuilt
           (since fill info is cached in styleblocks)%. Note that it calls STYLESHEET.CHANGE.SELECTIONS, since changing the fill to
           or from MULTI requires changing the format in which selections are represented.)

           (* * Fill in defaulted arguments.)

    [OR NEWFILL (SETQ NEWFILL (STYLE.PROP STYLESHEET 'NEED.NOT.FILL.IN]
    [OR BLOCKS (SETQ BLOCKS (STYLE.PROP STYLESHEET '\STYLE.BLOCKS]

           (* * update fill info in styleblocks.)

    [for BLOCK in BLOCKS as N from 1 do (LET ((FILL (if (LISTP NEWFILL)
                                                       then (CAR (FNTH NEWFILL N))
                                                     else NEWFILL)))

           (* * Record new fill type. Note if new fill type is just a single atom rather than a list, that atom applies to all items in the
           stylesheet.)

                                                (replace (STYLEBLOCK FILL) of BLOCK with FILL)

           (* * Build ALL-CLEAR submenu if necessary%: Menus that need not have any selections are always equipped with a
           CLEAR button in the submenu (not strictly necessary, since pressing a selected item on such a menu will deselect that item;
           but it serves as a simple visual aid to the user to tell him he need not make any selections in this menu);
           menus that can have multiple selections are equipped with an ALL button in the submenu
           (same reason as for the CLEAR button); and other menus have no submenu.)

                                                (replace (STYLEBLOCK SUBMENU) of BLOCK
                                                   with (SELECTQ FILL
                                                             (MULTI (create MENU
                                                                            ITEMS _ '(ALL CLEAR)
                                                                            WHENSELECTEDFN _ (FUNCTION
                                                                                                STYLESHEET.CLEAR.WHENSELECTEDFN
                                                                                                )
                                                                            MENUUSERDATA _ (fetch (STYLEBLOCK MENU)
                                                                                              of BLOCK)))
                                                             (T (create MENU
                                                                        ITEMS _ '(CLEAR)
                                                                        WHENSELECTEDFN _ (FUNCTION
                                                                                           STYLESHEET.CLEAR.WHENSELECTEDFN
                                                                                           )
                                                                        MENUUSERDATA _ (fetch (STYLEBLOCK MENU)
                                                                                          of BLOCK)))
                                                             (NIL NIL)
                                                             (\ILLEGAL.ARG NEWFILL]

           (* * Build mapping from submenus to styleblocks, which will enable the WHENSELECTEDFN of the submenu to get hold of
           the corresponding styleblock to modify its selections as necessary.)
```

```
    (STYLE.PROP STYLESHEET '\SUBMENU.TO.BLOCK (for BLOCK in BLOCKS bind SUBMENU
                                                 when (SETQ SUBMENU (fetch (STYLEBLOCK SUBMENU) of BLOCK))
                                                 collect (CONS SUBMENU BLOCK)))

        (* * Change selection format%: MULTI requires a list of selections, T or NIL require a single selection.)

    (STYLESHEET.CHANGE.SELECTIONS STYLESHEET NIL BLOCKS])
```

## (STYLESHEET.CHANGE.ITEMS

```
  [LAMBDA (STYLESHEET NEWITEMS)                                    ; Edited 26-Jan-88 17:56 by Trigg

;;; Rebuilds all the styleblocks.  Should be called whenever the user changes the items of a stylesheet.
    ;; rht 1/26/88: Now computes Done/Reset/Abort styleblock here and stashes on STYLEPROP rather than using globalvar.
    (LET [(BLOCKS (for MENU in NEWITEMS collect (create STYLEBLOCK
                                                       MENU _ MENU]
          (STYLE.PROP STYLESHEET '\STYLE.BLOCKS BLOCKS)
          [STYLE.PROP STYLESHEET '\DONE.STYLEBLOCK (create STYLEBLOCK
                                                           MENU _ (create MENU
                                                                          ITEMS _ '(DONE RESET ABORT)
                                                                          WHENSELECTEDFN _ (FUNCTION
                                                                                             STYLESHEET.DONE.FN]
        ;; Build mapping from menus to styleblocks.  This will allow the menus' WHENSELECTEDFN to get hold of the appropriate styleblock to
        ;; change its selection information.
        (STYLE.PROP STYLESHEET '\MENU.TO.BLOCK (for BLOCK in BLOCKS collect (CONS (fetch (STYLEBLOCK MENU)
                                                                                     of BLOCK)
                                                                                  BLOCK)))
        ;; Fill in fill info and selections (STYLESHEET.CHANGE.FILL calls STYLESHEET.CHANGE.SELECTIONS)
        (STYLESHEET.CHANGE.FILL STYLESHEET NIL BLOCKS)
        ;; Fill in item titles.
        (STYLESHEET.CHANGE.TITLES STYLESHEET NIL BLOCKS])
```

## (STYLESHEET.CHANGE.SELECTIONS

```
  [LAMBDA (STYLESHEET NEWSELECTIONS BLOCKS)                        (* hts%: "22-Mar-85 13:45")

        (* * Records new default selections in the styleblocks.)

        (* * Fill in defaulted arguments.)

    [OR NEWSELECTIONS (SETQ NEWSELECTIONS (STYLE.PROP STYLESHEET 'SELECTIONS]
    [OR BLOCKS (SETQ BLOCKS (STYLE.PROP STYLESHEET '\STYLE.BLOCKS]

        (* * Normalize selections and stick them into styleblocks. Selections must be normalized in two ways%: abbreviations for a
        menu item must be replaced by the entire menu item; and items with fill = MULTI should have a list of selections, but other
        items should have just a single selection (or NIL))

    (for BLOCK in BLOCKS as N from 1
       do (replace (STYLEBLOCK SELECTIONS) of BLOCK
             with (LET [(MENU (fetch (STYLEBLOCK MENU) of BLOCK))
                        (SELECTIONS (CAR (FNTH NEWSELECTIONS N]
                       (SELECTQ (fetch (STYLEBLOCK FILL) of BLOCK)
                            (MULTI (SETQ SELECTIONS (MKLIST SELECTIONS))
                                   (for MULTISEL in [LET [(FULL.SELECTIONS (for SUBSEL in SELECTIONS
                                                                              collect (STYLESHEET.MENUITEM SUBSEL
                                                                                                           MENU]
                                                                         (if (for SUBSEL in FULL.SELECTIONS always SUBSEL)
                                                                             then FULL.SELECTIONS
                                                                           else (LIST (STYLESHEET.MENUITEM SELECTIONS MENU]
                                                    when MULTISEL collect MULTISEL))
                            ((T NIL)
                                 (STYLESHEET.MENUITEM SELECTIONS MENU))
                            (SHOULDNT])
```

## (STYLESHEET.CHANGE.TITLES

```
  [LAMBDA (STYLESHEET NEWTITLES BLOCKS)                            (* hts%: "22-Mar-85 13:49")

        (* * Fills in item titles in styleblocks)

        (* * FIll in defaulted args.)

    [OR NEWTITLES (SETQ NEWTITLES (STYLE.PROP STYLESHEET 'ITEM.TITLES]
    [OR BLOCKS (SETQ BLOCKS (STYLE.PROP STYLESHEET '\STYLE.BLOCKS]

        (* * Fill in titles in styleblocks. ((CAR (FNTH xxx)) stuff ensures that if the title list is too short, titles in leftover styleblocks will
        be NILLed apropriately.))

    (for BLOCK in BLOCKS as N from 1 do (replace (STYLEBLOCK TITLE) of BLOCK with (CAR (FNTH NEWTITLES N])
```

## (STYLESHEET.MENUITEM

```
  [LAMBDA (SELECTION MENU)                                          (* hts%: "22-Mar-85 13:50")

          (* * Finds the full menu item corresponding to the given abbreviation.)

    (for MENUITEM in (fetch (MENU ITEMS) of MENU) thereis (OR (EQUAL MENUITEM SELECTION)
                                                             (STREQUAL (MKSTRING MENUITEM)
                                                                    (MKSTRING SELECTION))
                                                             (AND (LISTP MENUITEM)
                                                                  (STREQUAL (MKSTRING (CAR MENUITEM))
                                                                         (MKSTRING SELECTION])

)

(DEFINEQ

(STYLESHEET.SETUP
  [LAMBDA (STYLESHEET)                                              (* hts%: "22-Mar-85 14:05")

          (* * Changes the WHENSELECTEDFNs of all the menus to be the appropriate one for stylesheet menus, and returns the
          old WHENSELECTEDFNs. Also NILLs the SHADEDITEMS of each menu, since STYLESHEET.SHADE.SELECTIONS
          depends on the validity of this field. (ADDMENU should really do it --
          submit AR.))

    (for BLOCK in (STYLE.PROP STYLESHEET '\STYLE.BLOCKS) collect (LET ((MENU (fetch (STYLEBLOCK MENU) of BLOCK)))
                                                                     (PROG1 (fetch (MENU WHENSELECTEDFN)
                                                                               of MENU)
                                                                         (replace (MENU WHENSELECTEDFN)
                                                                            of MENU with (FUNCTION
                                                                                          STYLESHEET.WHENSELECTEDFN
                                                                                          ))
                                                                         (replace (MENU SHADEDITEMS) of MENU
                                                                            with NIL))])

(STYLESHEET.WAIT.TILL.DONE
  [LAMBDA (W STYLESHEET)                                            (* hts%: "21-Mar-85 18:39")

          (* * Wait until the user has filled everything in he needs to, and has hit the DONE button.)

          (* * make sure there is a mouse process running.)

    (SPAWN.MOUSE)
    (bind QUIT.TYPE do

          (* * keep the window on top to bring the users attention to it and to keep it from being covered.)

                    (TOTOPW W)
                    (WINDOWPROP W 'HOW NIL)
                    (WINDOWPROP W 'DONE (CREATE.EVENT 'DONE))
                    (AWAIT.EVENT (WINDOWPROP W 'DONE)
                           1000)
       repeatuntil (if (SETQ QUIT.TYPE (WINDOWPROP W 'HOW))
                       then                                         (* if not this was a timeout to bring the window back to the top.)
                            [if (EQ QUIT.TYPE 'ABORT)
                                then                                (* user selected the abort button.)
                                     T
                              else

          (* wait until the user hits the "done" button AND all menus have been selected from.)

                                   (for BLOCK in (STYLE.PROP STYLESHEET '\STYLE.BLOCKS)
                                      always (OR (fetch (STYLEBLOCK FILL) of BLOCK)
                                                 (fetch (STYLEBLOCK SELECTIONS) of BLOCK]
                       else NIL])

(STYLESHEET.GET.SELECTIONS
  [LAMBDA (STYLESHEET)                                              (* hts%: "22-Mar-85 14:09")

          (* * Gathers up the selections the user made, and records them as the new default selections for the stylesheet, so that if it
          is reused (and the default selections are not changed in between)%, the user will see his last selections.)

    (LET [(SELECTIONS (for BLOCK in (STYLE.PROP STYLESHEET '\STYLE.BLOCKS)
                         collect (LET ((SEL (fetch (STYLEBLOCK SELECTIONS) of BLOCK)))
                                      (SELECTQ (fetch (STYLEBLOCK FILL) of BLOCK)
                                           (MULTI (for SUBSEL in SEL
                                                     collect (if (AND (LISTP SUBSEL)
                                                                      (LISTP (CDR SUBSEL)))
                                                                 then (CADR SUBSEL)
                                                               else SUBSEL)))
                                           ((T NIL)
                                                (if (AND (LISTP SEL)
                                                         (LISTP (CDR SEL)))
                                                    then (CADR SEL)
                                                  else SEL))
                                           (SHOULDNT]
          (STYLE.PROP STYLESHEET 'SELECTIONS SELECTIONS)
```

```
         SELECTIONS])
```

## (STYLESHEET.CLEANUP

```
  [LAMBDA (W STYLESHEET OLD.WHENSELECTEDFNS)                    (* hts%: "22-Mar-85 14:10")

          (* * cleans up the WHENSELECTEDFNs in a stylesheet. And closes its window.)

    (for I in (STYLE.PROP STYLESHEET 'ITEMS) as W in OLD.WHENSELECTEDFNS
       do
          (* * Restore the old WHENSELECTEDFNs and MENUUSERDATAs.)

          (replace WHENSELECTEDFN of I with W))

          (* * Get rid of the stylesheet window)

    (CLOSEW W])

)

(DEFINEQ
```

## (STYLESHEET.WHENSELECTEDFN

```
  [LAMBDA (ELEMENT MENU BUTTON)                                 (* hts%: "22-Mar-85 14:11")

          (* * Special whenselectedfn for menus inside stylesheets. Permanently shades the selected item and records the new
          selection.)

    (LET* ([BLOCK (CDR (FASSOC MENU (STYLE.PROP (WINDOWPROP (WFROMMENU MENU)
                                                            'STYLESHEET)
                                                '\MENU.TO.BLOCK]
           (SELECTIONS (fetch (STYLEBLOCK SELECTIONS) of BLOCK)))

          (* * Modify recorded selections.)

          (replace (STYLEBLOCK SELECTIONS) of BLOCK with (SELECTQ (fetch (STYLEBLOCK FILL) of BLOCK)
                                                                  (T

          (* * Can have 0 or 1 selection. If selected same one twice, undo the selection, else change selection.)

                                                                     (if (EQ ELEMENT SELECTIONS)
                                                                         then NIL
                                                                       else ELEMENT))
                                                                  (MULTI

          (* * Can have any number of selection. If made same selection twice, remove it;
          else add it to the list of selections so far.)

                                                                        (if (FMEMB ELEMENT SELECTIONS)
                                                                            then (REMOVE ELEMENT SELECTIONS)
                                                                          else (CONS ELEMENT SELECTIONS)))
                                                                  (NIL

          (* * Must have exactly one selection. Change the current selection.)

                                                                       ELEMENT)
                                                                  (SHOULDNT)))

          (* * Display new selections.)

          (STYLESHEET.SHADE.SELECTIONS BLOCK])
```

## (STYLESHEET.CLEAR.WHENSELECTEDFN

```
  [LAMBDA (ELEMENT CLEAR.MENU BUTTON)                           (* hts%: "22-Mar-85 14:14")

          (* * WHENSELECTEDFN for the ALL-CLEAR submenus. Finds the styleblock corresponding to this submenu.
          If the user punched CLEAR, deselects all items in that styleblock's menu;
          if the user punched ALL, selects all the items in the menu. Changes shading to reflect new selections.)

    (LET [(BLOCK (CDR (FASSOC CLEAR.MENU (STYLE.PROP (WINDOWPROP (WFROMMENU CLEAR.MENU)
                                                                 'STYLESHEET)
                                                     '\SUBMENU.TO.BLOCK]
          (replace (STYLEBLOCK SELECTIONS) of BLOCK with (SELECTQ ELEMENT
                                                                  (CLEAR NIL)
                                                                  (ALL (for MENUITEM
                                                                          in (fetch (MENU ITEMS)
                                                                                of (fetch (STYLEBLOCK MENU) of BLOCK))
                                                                          collect MENUITEM))
                                                                  NIL))
          (STYLESHEET.SHADE.SELECTIONS BLOCK])
```

## (STYLESHEET.DONE.FN

```
  [LAMBDA (ITEM M BUTTON)                                       (* hts%: "21-Mar-85 16:39")

          (* * WHENSELECTEDFN for the DONE-ABORT-RESET menu.)
```

```
    (LET ((W (WFROMMENU M)))
         (SELECTQ ITEM
              ((DONE ABORT)

          (* * Done selecting from this stylesheet. Notify calling process.)

                  (WINDOWPROP W 'HOW ITEM)
                  (NOTIFY.EVENT (WINDOWPROP W 'DONE)))
             (RESET (LET [(STYLESHEET (WINDOWPROP W 'STYLESHEET]

          (* * Restore the original selections.)

                         (STYLE.PROP STYLESHEET 'SELECTIONS (STYLE.PROP STYLESHEET 'SELECTIONS))

          (* * Shade the original selections.)

                         (for BLOCK in (STYLE.PROP STYLESHEET '\STYLE.BLOCKS) do (STYLESHEET.SHADE.SELECTIONS
                                                                                  BLOCK))))
             NIL])

)

(DEFINEQ
```

## (STYLESHEET.ITEM.HEIGHT

```
  [LAMBDA (BLOCK TITLEFONT)                                          (* hts%: "20-Mar-85 19:50")

          (* * Returns the height in pixels the given block would occupy in a stylesheet.)

    (PLUS (if (fetch (STYLEBLOCK TITLE) of BLOCK)
              then (PLUS 2 (FONTHEIGHT TITLEFONT))
            else 0)
          (fetch (MENU IMAGEHEIGHT) of (fetch (STYLEBLOCK MENU) of BLOCK))
          (LET ((SUBMENU (fetch (STYLEBLOCK SUBMENU) of BLOCK)))
               (if SUBMENU
                   then (PLUS 2 (fetch (MENU IMAGEHEIGHT) of SUBMENU))
                 else 0])
```

## (STYLESHEET.ITEM.WIDTH

```
  [LAMBDA (BLOCK TITLEFONT)                                          (* hts%: "20-Mar-85 19:59")

          (* * returns the width in piexels that the given block would occupy were it printed in a stylesheet.)

    (MAX (LET ((TITLE (fetch (STYLEBLOCK TITLE) of BLOCK)))
              (if TITLE
                  then (STRINGWIDTH TITLE TITLEFONT)
                else 0))
         (fetch (MENU IMAGEWIDTH) of (fetch (STYLEBLOCK MENU) of BLOCK))
         (LET ((SUBMENU (fetch (STYLEBLOCK SUBMENU) of BLOCK)))
              (if SUBMENU
                  then (fetch (MENU IMAGEWIDTH) of SUBMENU)
                else 0])

)

(DEFINEQ
```

## (STYLESHEET.CREATE.WINDOW

```
  [LAMBDA (STYLESHEET)                                               (* hts%: "22-Mar-85 14:18")

          (* * Lay out stylesheet window of appropriate size and fill it in.)

    (LET ((POS (STYLE.PROP STYLESHEET 'POSITION))
          (TITLE (STYLE.PROP STYLESHEET 'TITLE))
          (HEIGHT (STYLESHEET.IMAGEHEIGHT STYLESHEET))
          (WIDTH (STYLESHEET.IMAGEWIDTH STYLESHEET)))

          (* * If position for stylesheet is not provided, prompt for it.)

         (if (NULL POS)
             then (SETQ POS (GETBOXPOSITION WIDTH HEIGHT)))

          (* * Ensure that stylesheet is on the screen.)

         [replace XCOORD of POS with (MAX 1 (MIN (fetch XCOORD of POS)
                                                 (IDIFFERENCE SCREENWIDTH WIDTH]
         [replace YCOORD of POS with (MAX 1 (MIN (fetch YCOORD of POS)
                                                 (IDIFFERENCE SCREENHEIGHT HEIGHT]

          (* * Lay out a window big enough to fit all the items.)

         (LET ((W (CREATEW (CREATEREGION (fetch XCOORD of POS)
                                         (fetch YCOORD of POS)
                                         WIDTH HEIGHT)
                           TITLE)))
```

```
        (* * Save the stylesheet on its window, where it can be accessed by the WHENSELECTEDFNS etc.
        running in a different process.)

            (WINDOWPROP W 'STYLESHEET STYLESHEET)

        (* * Give the window a REPAINTFN so it can be redisplayed properly.)

            (WINDOWPROP W 'REPAINTFN (FUNCTION STYLESHEET.FILL.IN.WINDOW))

        (* * Fill in the window.)

            (REDISPLAYW W)
            W])
```

## (STYLESHEET.FILL.IN.WINDOW

```
  [LAMBDA (W)                                                    ; Edited 26-Jan-88 17:56 by Trigg

;;; Put items into stylesheet and shade default menu selections.
    ;; rht 1/26/88: Now gets Done/Reset/Abort styleblock off of STYLEPROP rather than using globalvar.
    (for M in (WINDOWPROP W 'MENU) do (DELETEMENU M NIL W))
    (LET [(STYLESHEET (WINDOWPROP W 'STYLESHEET]
         (bind (TITLEFONT _ (STYLE.PROP STYLESHEET 'ITEM.TITLE.FONT))
               (HEIGHT _ (WINDOWPROP W 'HEIGHT))
               (XOFFSET _ 0)
               WIDTH for BLOCK in [APPEND (STYLE.PROP STYLESHEET '\STYLE.BLOCKS)
                                          (LIST (STYLE.PROP STYLESHEET '\DONE.STYLEBLOCK]
            do (SETQ WIDTH (STYLESHEET.ITEM.WIDTH BLOCK TITLEFONT))
               (STYLESHEET.ADD.MENU BLOCK W XOFFSET HEIGHT WIDTH TITLEFONT)
               (SETQ XOFFSET (PLUS XOFFSET WIDTH 2])
```

## (STYLESHEET.ADD.MENU

```
  [LAMBDA (BLOCK W XOFFSET HEIGHT WIDTH TITLEFONT)               (* hts%: "22-Mar-85 14:24")

        (* * Adds the current styleblock (title, menu, and submenu) to the stylesheet.
        XOFFSET determines the placement of left boundary of the styleblock.
        HEIGHT and WIDTH bound the styleblock above and to the right.
        Centers the pieces of the styleblock within this box.)

    (LET ((TOP HEIGHT))

        (* * Title stuff)

        [LET ((TITLE (fetch (STYLEBLOCK TITLE) of BLOCK)))
             (if TITLE
                 then (MOVETO (IPLUS XOFFSET (IQUOTIENT (DIFFERENCE WIDTH (STRINGWIDTH TITLE TITLEFONT))
                                                        2))
                              [DIFFERENCE TOP (DIFFERENCE (FONTPROP TITLEFONT 'HEIGHT)
                                                          (FONTPROP TITLEFONT 'DESCENT]
                              W)
                      (printout W .FONT TITLEFONT TITLE)
                      (SETQ TOP (DIFFERENCE TOP (PLUS (FONTHEIGHT TITLEFONT)
                                                      2]

        (* * Stick the menu on the stylesheet window.)

        [LET ((MENU (fetch (STYLEBLOCK MENU) of BLOCK)))
             [ADDMENU MENU W (create POSITION
                                     XCOORD _ (IPLUS XOFFSET (IQUOTIENT (DIFFERENCE WIDTH (fetch (MENU IMAGEWIDTH
                                                                                                 )
                                                                                          of MENU))
                                                                        2))
                                     YCOORD _ (IDIFFERENCE TOP (fetch IMAGEHEIGHT of MENU]
             (SETQ TOP (DIFFERENCE TOP (PLUS (fetch IMAGEHEIGHT of MENU)
                                             2]

        (* * Shade in selections.)

        (STYLESHEET.SHADE.SELECTIONS BLOCK)

        (* * Add clear/all menu at bottom of this item.)

        [LET ((SUBMENU (fetch (STYLEBLOCK SUBMENU) of BLOCK)))
             (if SUBMENU
                 then (ADDMENU SUBMENU W (create POSITION
                                                 XCOORD _ (IPLUS XOFFSET (IQUOTIENT (DIFFERENCE
                                                                                        WIDTH
                                                                                        (fetch (MENU IMAGEWIDTH)
                                                                                           of SUBMENU))
                                                                                    2))
                                                 YCOORD _ (IDIFFERENCE TOP (fetch IMAGEHEIGHT of SUBMENU]
        NIL])
```

## (STYLESHEET.SHADE.SELECTIONS

```
  [LAMBDA (BLOCK)                                          (* hts%: "22-Mar-85 14:26")
```

(* * Updates the selections shaded on the screen to reflect those recorded internally for the specified Makes use of the SHADEDITEMS field of the menu to tell what has already been shaded.
Format of SHADEDITEMS is an alist mapping the number of the menu item onto its shade.)

```
    (LET* ((MENU (fetch (STYLEBLOCK MENU) of BLOCK))
           (SHADEDITEMS (fetch (MENU SHADEDITEMS) of MENU))
           (SELECTIONS (fetch (STYLEBLOCK SELECTIONS) of BLOCK)))
          (if (NEQ (fetch (STYLEBLOCK FILL) of BLOCK)
                   'MULTI)
              then (SETQ SELECTIONS (LIST SELECTIONS)))
          (for MENUITEM in (fetch (MENU ITEMS) of MENU) as ITEMNUMBER from 1
             do (LET* [(SELECTED (FMEMB MENUITEM SELECTIONS))
                       (SHADENTRY (FASSOC ITEMNUMBER SHADEDITEMS))
                       (SHADED (AND (LISTP SHADENTRY)
                                    (NEQ (CDR SHADENTRY)
                                         0]
                      (if (AND SHADED (NOT SELECTED))
                          then (SHADEITEM MENUITEM MENU WHITESHADE)
                        elseif (AND SELECTED (NOT SHADED))
                          then (SHADEITEM MENUITEM MENU STYLESHEET.SELECTED.SHADE])

)

(DEFINEQ
```

## (STYLESHEET.AT.LOAD

```
  [LAMBDA NIL                                              ; Edited 26-Jan-88 11:51 by Trigg
```

;;; Sets up global variables for the stylesheet package.

;; rht 1/26/88: No longer computes STYLESHEET.DONE.MENU globalvar here. It gets computed each time STYLESHEET is called.

```
    (SETQ STYLESHEET.SELECTED.SHADE BLACKSHADE])

)
```

(**STYLESHEET.AT.LOAD**)

```
(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS STYLESHEET.SELECTED.SHADE)
)

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

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA STYLE.PROP CREATE.STYLE)
)

(PUTPROPS STYLESHEET COPYRIGHT ("Xerox Corporation" 1983 1985 1987 1988))
```

## FUNCTION INDEX

CREATE.STYLE .....1
STYLE.PROP .....2
STYLESHEET .....2
STYLESHEET.ADD.MENU .....8
STYLESHEET.AT.LOAD .....9
STYLESHEET.CHANGE.FILL .....3
STYLESHEET.CHANGE.ITEMS .....4
STYLESHEET.CHANGE.SELECTIONS .....4
STYLESHEET.CHANGE.TITLES .....4
STYLESHEET.CLEANUP .....6
STYLESHEET.CLEAR.WHENSELECTEDFN ..6
STYLESHEET.CREATE.WINDOW .....7
STYLESHEET.DONE.FN .....6
STYLESHEET.FILL.IN.WINDOW .....8
STYLESHEET.GET.SELECTIONS .....5
STYLESHEET.IMAGEHEIGHT .....2
STYLESHEET.IMAGEWIDTH .....2
STYLESHEET.ITEM.HEIGHT .....7
STYLESHEET.ITEM.WIDTH .....7
STYLESHEET.MENUITEM .....4
STYLESHEET.SETUP .....5
STYLESHEET.SHADE.SELECTIONS .....9
STYLESHEET.WAIT.TILL.DONE .....5
STYLESHEET.WHENSELECTEDFN .....6
STYLESHEETP .....1

## RECORD INDEX

STYLEBLOCK .....3

## PROPERTY INDEX

STYLESHEET .....1