File created: 16-Mar-2024 12:08:24 **{DSK}<home>larry>il>medley>lispusers>WHO-LINE.;6**

edit by: lmm

changes to: (VARS WHO-LINECOMS)

previous date: 15-Mar-2024 07:19:58 {DSK}<home>larry>il>medley>lispusers>WHO-LINE.;3

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ WHO-LINECOMS
       (
```

;;; Define a WHO-LINE window that displays the current state of a number of user specified attributes.

```
        ;;
        ;; Public fn for manipulating the who-line
        (FNS INSTALL-WHO-LINE-OPTIONS)
```

;;; --

;;; Some fns that compute useful values for the who-line, and act as nice button event fns

```
        ;;
        ;; Showing / changing the current logged in user
        (FNS WHO-LINE-USERNAME WHO-LINE-CHANGE-USER WHO-LINE-USER-AFTER-LOGIN)
        (VARIABLES *WHO-LINE-CURRENT-USER* *WHO-LINE-USER-ENTRY*)
        (DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE (ADDVARS (\SYSTEMCACHEVARS *WHO-LINE-CURRENT-USER*)
                                                           (\AFTERLOGINFNS WHO-LINE-USER-AFTER-LOGIN)))
        ;;
        ;; Showing the current machine name
        (FNS WHO-LINE-HOST-NAME)
        (VARIABLES *WHO-LINE-HOST-NAME* *WHO-LINE-HOST-NAME-ENTRY*)
        (DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE (ADDVARS (\SYSTEMCACHEVARS *WHO-LINE-HOST-NAME*)))
        ;;
        ;; Showing / changing the current tty process package
        (FNS CURRENT-TTY-PACKAGE SET-PACKAGE-INTERACTIVELY SET-TTY-PACKAGE-INTERACTIVELY)
        (VARIABLES *WHO-LINE-PACKAGE-NAME-CACHE* *WHO-LINE-PACKAGE-ENTRY*)
        ;;
        ;; Showing / changing the current tty process readtable
        (FNS CURRENT-TTY-READTABLE-NAME SET-READTABLE-INTERACTIVELY SET-TTY-READTABLE-INTERACTIVELY)
        (VARIABLES *WHO-LINE-READTABLE-ENTRY*)
        ;;
        ;; Showing / changing the current tty process
        (FNS WHO-LINE-TTY-PROCESS CHANGE-TTY-PROCESS-INTERACTIVELY)
        (VARIABLES *WHO-LINE-TTY-PROC-ENTRY*)
        ;;
        ;; Showing / changing the currently connected directory
        (FNS WHO-LINE-CURRENT-DIRECTORY SET-CONNECTED-DIRECTORY-INTERACTIVELY)
        (VARIABLES *WHO-LINE-DIRECTORIES* *WHO-LINE-LAST-DIRECTORY* *WHO-LINE-DIRECTORY-ENTRY*)
        ;;
        ;; Showing / changing the current VMem utilization
        (FNS WHO-LINE-VMEM WHO-LINE-SAVE-VMEM)
        (VARIABLES *WHO-LINE-LAST-VMEM* *WHO-LINE-VMEM-ENTRY*)
        (DECLARE%: EVAL@COMPILE DONTCOPY DONTEVAL@LOAD (FILES (LOADCOMP)
                                                              LLFAULT MODARITH)
               (P (CHECKIMPORTS '(LLPARAMS)
                        T)))
        ;;
        ;; Showing the percent of symbol-space currently used
        (FUNCTIONS WHO-LINE-SYMBOL-SPACE)
        (VARIABLES *WHO-LINE-SYMBOL-SPACE* *WHO-LINE-SYMBOL-SPACE-ENTRY*)
        ;;
        ;; Showing the current time
        (FNS WHO-LINE-TIME WHO-LINE-SET-TIME)
        (VARIABLES *WHO-LINE-TIMER* *WHO-LINE-OLD-TIME* *WHO-LINE-TIME-ENTRY*)
```

```
        [DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE (APPENDVARS (\SYSTEMTIMERVARS (*WHO-LINE-TIMER* SECONDS]

;;; ------------------------------------------------------------------------

;;; Some as yet un-debugged entries.  Try at your own risk.

        ;;
        ;; Showing the machine-active entry
        (FNS WHO-LINE-SHOW-ACTIVE \UPDATE-WHO-LINE-ACTIVE-FLAG \PERIODICALLY-WHO-LINE-SHOW-ACTIVE)
        (VARIABLES *WHO-LINE-ACTIVE-PERIOD* *WHO-LINE-ACTIVE-TIMER* *WHO-LINE-SHOW-ACTIVE-ENTRY*)
        [DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE (APPENDVARS (\SYSTEMTIMERVARS (*WHO-LINE-ACTIVE-TIMER*
                                                                                 MILLISECONDS]
        ;;
        ;; Showing / changing the current reader profile
        (FNS CURRENT-PROFILE SET-PROFILE-INTERACTIVELY SET-TTY-PROFILE-INTERACTIVELY)
        (VARIABLES *WHO-LINE-PROFILE-ENTRY*)
        ;;
        ;; Showing the state of the current TTY process
        (FNS WHO-LINE-TTY-STATE WHO-LINE-WHAT-IS-RUNNING)
        (VARIABLES *WHO-LINE-STATE* *WHO-LINE-STATE-UNINTERESTING-FNS* *WHO-LINE-TTY-STATE-ENTRY*)
        (PROP WHO-LINE-STATE AWAIT.EVENT BLOCK EXCHANGEPUPS GETPUP SENDPUP WAIT.FOR.TTY \TTYBACKGROUND
              \WAITFORSYSBUFP \\getkey \SENDLEAF PUTSEQUIN \LEAF.READPAGES)

;;; ------------------------------------------------------------------------

;;; Default options for the who-line

        (VARIABLES *WHO-LINE-ENTRIES* *WHO-LINE-ENTRY-REGISTRY* *WHO-LINE-ANCHOR* *WHO-LINE-NAME-FONT*
               *WHO-LINE-VALUE-FONT* *WHO-LINE-DISPLAY-NAMES?* *WHO-LINE-COLOR* *WHO-LINE-TITLE*
               *WHO-LINE-BORDER* *WHO-LINE-UPDATE-INTERVAL*)

;;; ------------------------------------------------------------------------

;;; Internal fns

        (FNS REDISPLAY-WHO-LINE PERIODICALLY-UPDATE-WHO-LINE SETUP-WHOLINE-TIMER UPDATE-WHO-LINE
             WHEN-WHO-LINE-SELECTED-FN WHO-LINE-CONTROL-SELECT WHO-LINE-COPY-INSERT)
        (FNS WHO-LINE-REDISPLAY-INTERRUPT)
        (VARIABLES *WHO-LINE* *WHO-LINE-UPDATE-TIMER*)
        [DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE (APPENDVARS (AFTERSYSOUTFORMS (INSTALL-WHO-LINE-OPTIONS))
                                                              (AFTERMAKESYSFORMS (INSTALL-WHO-LINE-OPTIONS))
                                                              (\SYSTEMTIMERVARS (*WHO-LINE-UPDATE-TIMER* TICKS]
        (FUNCTIONS INVERT-WHO-LINE-ENTRY)
        (DECLARE%: DONTCOPY (RECORDS WHO-LINE-ENTRY))
                                                                  ; Macros that lets us lock down the Who-Line while we evaluate
                                                                  ; some forms
        (FUNCTIONS WITH-WHO-LINE WITH-AVAILABLE-WHO-LINE)

;;; ------------------------------------------------------------------------

;;; Initialize the who-line

        (DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE (P (INSTALL-WHO-LINE-OPTIONS))
               (ADDVARS (BACKGROUNDFNS PERIODICALLY-UPDATE-WHO-LINE)))

;;; ------------------------------------------------------------------------

;;; Filemanager stuff

        (DECLARE%: DONTCOPY (PROP MAKEFILE-ENVIRONMENT WHO-LINE)
               (PROP FILETYPE WHO-LINE))))

;;; Define a WHO-LINE window that displays the current state of a number of user specified attributes.
;;
;; Public fn for manipulating the who-line
```

```
(DEFINEQ

(INSTALL-WHO-LINE-OPTIONS
  [LAMBDA NIL                                          ; Edited 16-May-88 14:19 by smL

;;;

;;; Install new descriptions of the values to be displayed in the who-line.

;;; Each description is a list of four items:  then name of the value, a form that will compute the value, the maximum number of characters in the resulting
;;; value, and an optional function that will be FUNCALLed if/when that item is moused in the who-line.

;;;
    ;;
    ;; Create the who-line window if it isn't there already
    ;;
    (if (NOT (AND (BOUNDP '*WHO-LINE*)
                  (WINDOWP *WHO-LINE*)))
        then (SETQ *WHO-LINE* (CREATEW (CREATEREGION 0 0 100 20)
                                       NIL NIL T))
             (WINDOWPROP *WHO-LINE* 'LOCK (CREATE.MONITORLOCK "WHO-LINE")))
    (WITH-WHO-LINE *WHO-LINE* (WINDOWPROP *WHO-LINE* 'VALID NIL)
           (OPENW *WHO-LINE*)
           (LET ((CURRENT-LEFT 0)
                 ENTRIES)
                ;;
                ;; Make sure the who-line has all the correct window properties
                ;;
                (WINDOWPROP *WHO-LINE* 'REPAINTFN 'REDISPLAY-WHO-LINE)
                (WINDOWPROP *WHO-LINE* 'BUTTONEVENTFN 'WHEN-WHO-LINE-SELECTED-FN)
                (WINDOWPROP *WHO-LINE* 'DISPLAY-NAMES? *WHO-LINE-DISPLAY-NAMES?*)
                (WINDOWPROP *WHO-LINE* 'ANCHOR *WHO-LINE-ANCHOR*)
                (WINDOWPROP *WHO-LINE* 'NAME-FONT *WHO-LINE-NAME-FONT*)
                (WINDOWPROP *WHO-LINE* 'VALUE-FONT *WHO-LINE-VALUE-FONT*)
                (WINDOWPROP *WHO-LINE* 'COLOR *WHO-LINE-COLOR*)
                (WINDOWPROP *WHO-LINE* 'TITLE *WHO-LINE-TITLE*)
                (WINDOWPROP *WHO-LINE* 'BORDER *WHO-LINE-BORDER*)
                (WINDOWPROP *WHO-LINE* 'UPDATE-INTERVAL (FIX (TIMES *WHO-LINE-UPDATE-INTERVAL* \RCLKMILLISECOND)
                                                             ))
                (SETQ *WHO-LINE-UPDATE-TIMER* (SETUP-WHOLINE-TIMER *WHO-LINE-UPDATE-TIMER*))
                ;;
                ;; Create and fill in the who-line entries that go on the window.
                ;; This entails computing the positions of the entries in the who-line
                ;;
                [SETQ ENTRIES (for ITEM in *WHO-LINE-ENTRIES* bind (DISPLAY-NAMES? _ (WINDOWPROP *WHO-LINE*
                                                                                                 'DISPLAY-NAMES?))
                                                                   (VALUE-FONT _ (WINDOWPROP *WHO-LINE*
                                                                                             'VALUE-FONT))
                                                                   (NAME-FONT _ (WINDOWPROP *WHO-LINE*
                                                                                            'NAME-FONT))
                                   collect (LET [(ENTRY (create WHO-LINE-ENTRY
                                                                NAME _ (CL:FIRST ITEM)
                                                                FORM _ (CL:SECOND ITEM]
                                                (with WHO-LINE-ENTRY ENTRY
                                                      ;;
                                                      ;; Leave a little space (the size of an "A") between the previous value and this name
                                                      ;;
                                                      (SETQ NAME-START (PLUS (STRINGWIDTH "A" VALUE-FONT)
                                                                             CURRENT-LEFT))
                                                      [if DISPLAY-NAMES?
                                                          then (SETQ CURRENT-LEFT (PLUS NAME-START (STRINGWIDTH NAME
                                                                                                                NAME-FONT]
                                                      ;;
                                                      ;; The value is displayed after the name, with a little space between them
                                                      ;;
                                                      (SETQ VALUE-START (PLUS CURRENT-LEFT (STRINGWIDTH "A"
                                                                                                        VALUE-FONT)))
                                                      [SETQ VALUE-END (PLUS VALUE-START (TIMES (CL:THIRD ITEM)
                                                                                               (STRINGWIDTH "A"
                                                                                                            VALUE-FONT]
                                                                  ; Leave a little extra space after each value
                                                      (SETQ CURRENT-LEFT (PLUS VALUE-END (STRINGWIDTH "A" VALUE-FONT
                                                                                                      )))
                                                      ;;
                                                      ;; Set the when-selected-fn
```

```
                                        ;;
                                        (SETQ WHEN-SELECTED-FN (CL:FOURTH ITEM))
                                        ;;
                                        ;; And the reset-form
                                        ;;
                                        (SETQ RESET-FORM (CL:FIFTH ITEM))
                                        ;;
                                        ;; And return the filled in entry
                                        ;;
                                        ENTRY]
                ;;
                ;; Reshape the window to hold the new in info
                ;;
                (LET [[HORIZ-ANCHOR (if (POSITIONP (WINDOWPROP *WHO-LINE* 'ANCHOR))
                                        then (fetch XCOORD of (WINDOWPROP *WHO-LINE* 'ANCHOR))
                                      else (OR [for anchor in (WINDOWPROP *WHO-LINE* 'ANCHOR)
                                                  thereis (MEMB anchor '(:LEFT :CENTER :JUSTIFY :RIGHT]
                                               (ERROR "No horizontal anchor specified" (WINDOWPROP *WHO-LINE*
                                                                                                   'ANCHOR]
                      [VERT-ANCHOR (if (POSITIONP (WINDOWPROP *WHO-LINE* 'ANCHOR))
                                       then (fetch YCOORD of (WINDOWPROP *WHO-LINE* 'ANCHOR))
                                     else (OR [for anchor in (WINDOWPROP *WHO-LINE* 'ANCHOR)
                                                 thereis (MEMB anchor '(:TOP :BOTTOM]
                                              (ERROR "No vertical anchor specified" (WINDOWPROP *WHO-LINE*
                                                                                                'ANCHOR]
                      [WIDTH (WIDTHIFWINDOW CURRENT-LEFT (WINDOWPROP *WHO-LINE* 'BORDER]
                      (HEIGHT (HEIGHTIFWINDOW (MAX (FONTPROP (WINDOWPROP *WHO-LINE* 'NAME-FONT)
                                                             'HEIGHT)
                                                   (FONTPROP (WINDOWPROP *WHO-LINE* 'VALUE-FONT)
                                                             'HEIGHT))
                                              (WINDOWPROP *WHO-LINE* 'TITLE)
                                              (WINDOWPROP *WHO-LINE* 'BORDER]
                ;;
                ;; Make sure the window fits on the screen (i.e. doesn't run off the edge, and is justified against left and right sides if the user
                ;; wants).
                ;; If the items don't fit, change the length of each item so they do.
                ;; Do this by distributing the "pain" among all the entries in the who-line.
                ;;
                (if (OR (GREATERP WIDTH SCREENWIDTH)
                        (EQ HORIZ-ANCHOR :JUSTIFY))
                    then (for ENTRY in ENTRIES bind (REMAINING-ADJUSTMENT _ (DIFFERENCE SCREENWIDTH WIDTH))
                                                    [REMAINING-VALUE-SIZE _
                                                           (for ENTRY in ENTRIES
                                                              sum (with WHO-LINE-ENTRY ENTRY (DIFFERENCE
                                                                                                 VALUE-END
                                                                                                 VALUE-START]
                                                    (RUNNING-ADJUSTMENT _ 0)
                                                    ENTRY-ADJUSTMENT
                              do (with WHO-LINE-ENTRY ENTRY
                                       ;;
                                       ;; Figure out how much this entry value gets adjusted.
                                       ;;
                                       ;; Note that, by keeping track of the remaing adjustment needed, we avoid problems with
                                       ;; round-off.
                                       ;;
                                       (SETQ ENTRY-ADJUSTMENT (QUOTIENT (TIMES REMAINING-ADJUSTMENT
                                                                               (DIFFERENCE VALUE-END
                                                                                           VALUE-START))
                                                                        REMAINING-VALUE-SIZE))
                                       ;;
                                       ;; Update this entry size & position
                                       ;;
                                       (add NAME-START RUNNING-ADJUSTMENT)
                                       (add VALUE-START RUNNING-ADJUSTMENT)
                                       (add RUNNING-ADJUSTMENT ENTRY-ADJUSTMENT)
                                       (add VALUE-END RUNNING-ADJUSTMENT))
                              finally (SETQ WIDTH SCREENWIDTH)))
                ;;
                ;; Set the who-line window size so it can't be reshaped
                ;;
                (WINDOWPROP *WHO-LINE* 'MAXSIZE (CONS WIDTH HEIGHT))
```

```
                    (WINDOWPROP *WHO-LINE* 'MINSIZE (CONS WIDTH HEIGHT))
                    ;;
                    ;; The anchor-point decribes where on the screen the who-line should be placed.
                    ;; The CAR should be one of :JUSTIFY, :LEFT, :RIGHT, or :CENTER.
                    ;; The CADR should be one of :TOP, :BOTTOM, or :CENTER.
                    ;;
                    (SHAPEW *WHO-LINE* (CREATEREGION (SELECTQ HORIZ-ANCHOR
                                                              ((:JUSTIFY :LEFT)
                                                                  0)
                                                              (:RIGHT (DIFFERENCE SCREENWIDTH WIDTH))
                                                              (:CENTER (QUOTIENT (DIFFERENCE SCREENWIDTH WIDTH)
                                                                                 2))
                                                              HORIZ-ANCHOR)
                                                     (SELECTQ VERT-ANCHOR
                                                          (:TOP (DIFFERENCE SCREENHEIGHT HEIGHT))
                                                          (:BOTTOM 0)
                                                          (:CENTER (QUOTIENT (DIFFERENCE SCREENHEIGHT HEIGHT)
                                                                             2))
                                                          VERT-ANCHOR)
                                                     WIDTH HEIGHT)))
                ;;
                ;; The values should be centered vertically between the top and the bottom of the window
                ;;
                (WINDOWPROP *WHO-LINE* 'VALUE-BOTTOM (PLUS (FONTPROP (WINDOWPROP *WHO-LINE* 'VALUE-FONT)
                                                                     'DESCENT)
                                                           (QUOTIENT (DIFFERENCE (WINDOWPROP *WHO-LINE*
                                                                                             'HEIGHT)
                                                                                 (FONTPROP (WINDOWPROP *WHO-LINE*
                                                                                                       'VALUE-FONT)
                                                                                           'HEIGHT))
                                                                     2)))
                ;; Cache a bitmap that is the same size as the inside of the who-line, and a display stream onto the bitmap.
                [WINDOWPROP *WHO-LINE* 'TEMP-STREAM (DSPCREATE (BITMAPCREATE (WINDOWPROP *WHO-LINE* 'WIDTH)
                                                                             (WINDOWPROP *WHO-LINE* 'HEIGHT]
                ;;
                ;; Install the entries
                ;;
                (WINDOWPROP *WHO-LINE* 'ENTRIES ENTRIES)
                ;;
                ;; Finally, update the window
                ;;
                (REDISPLAY-WHO-LINE *WHO-LINE*)
                (WINDOWPROP *WHO-LINE* 'VALID T])

)


;;; -------------------------------------------------------------------------

;;; Some fns that compute useful values for the who-line, and act as nice button event fns
;;
;; Showing / changing the current logged in user

(DEFINEQ

(WHO-LINE-USERNAME
  [LAMBDA NIL                                                    ; Edited 30-Jun-88 15:41 by smL

;;;

;;; Return the name of the currently logged in user.  Avoid consing up a new string if possible.

;;;
      ;; The cached value in *WHO-LINE-CURRENT-USER* gets invalidated by an entry on the list of \SYSTEMCACHEVARS, and by a function on the
      ;; list of \AFTERLOGINFNS
      ;;
      (DECLARE (GLOBALVARS *WHO-LINE-CURRENT-USER*))
      (if *WHO-LINE-CURRENT-USER*
          then *WHO-LINE-CURRENT-USER*
        else (SETQ *WHO-LINE-CURRENT-USER* (USERNAME NIL NIL T])

(WHO-LINE-CHANGE-USER
  [LAMBDA NIL                                                    (* smL "17-Nov-86 11:19")
```

```
;;;

;;; Change the currently logged in user

;;;

    (if [MENU (create MENU
                      TITLE _ "Change user?"
                      CENTERFLG _ T
                      ITEMS _ '(("Yes" T "Log in as a different user")
                                ("No" NIL "Don't change the current user"]
        then (LOGIN])

(WHO-LINE-USER-AFTER-LOGIN
  [LAMBDA (HOST USER)                                   ; Edited 30-Jun-88 15:34 by smL
    (CL:WHEN (NULL HOST)
           (SETQ *WHO-LINE-CURRENT-USER* NIL])

)


(DEFGLOBALVAR *WHO-LINE-CURRENT-USER* NIL
   "Cached name of the current logged in user")


(CL:DEFPARAMETER *WHO-LINE-USER-ENTRY* '("User" (WHO-LINE-USERNAME)
                                                10 WHO-LINE-CHANGE-USER (SETQ *WHO-LINE-CURRENT-USER* NIL)
                                                "Name of the currently logged in user")
                                       "Who-Line entry for displaying the name of the currently logged in
                                       user")

(DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE

(ADDTOVAR \SYSTEMCACHEVARS *WHO-LINE-CURRENT-USER*)

(ADDTOVAR \AFTERLOGINFNS WHO-LINE-USER-AFTER-LOGIN)
)

;;
;; Showing the current machine name

(DEFINEQ

(WHO-LINE-HOST-NAME
  [LAMBDA NIL                                           ; Edited 12-Apr-2023 22:09 by lmm
                                                        ; Edited 14-Jan-87 12:46 by smL

;;;

;;; Return the name of the curren workstation.  Avoid consing up a new string if possible.

;;;
    ;;
    ;; The cached value in *WHO-LINE-HOST-NAME* gets invalidated by an entry on the list of \SYSTEMCACHEVARS
    ;;
    (DECLARE (GLOBALVARS *WHO-LINE-HOST-NAME*))
    (IF *WHO-LINE-HOST-NAME*
        THEN *WHO-LINE-HOST-NAME*
      ELSE (SETQ *WHO-LINE-HOST-NAME* (UNIX-GETPARM "HOSTNAME"])

)


(DEFGLOBALVAR *WHO-LINE-HOST-NAME* NIL
   "Cached name of the current machine, for the Who-Line")


(CL:DEFPARAMETER *WHO-LINE-HOST-NAME-ENTRY* '("on" (WHO-LINE-HOST-NAME)
                                                   10 NIL (SETQ *WHO-LINE-HOST-NAME* NIL)
                                                   "Name of the currently running machine")
                                            "Who-Line entry for displaying the name of the current
                                            machine")

(DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE

(ADDTOVAR \SYSTEMCACHEVARS *WHO-LINE-HOST-NAME*)
)

;;
;; Showing / changing the current tty process package

(DEFINEQ
```

```
(CURRENT-TTY-PACKAGE
  [LAMBDA NIL                                                    ; Edited 17-Mar-87 17:52 by smL

;;;

;;; Return the name of the current package of the current TTY process

;;;

    (LET [(PACKAGE (PROCESS.EVALV (TTY.PROCESS)
                         '*PACKAGE*]
         ;;
         ;; The *WHO-LINE-PACKAGE-NAME-CACHE* AList is used to cache computed package names with terminating ":"s.
         ;; This lets us display the name with a colon w/o having to allocate new strings all the time.
         ;;
         (OR (CDR (ASSOC PACKAGE *WHO-LINE-PACKAGE-NAME-CACHE*))
             (PUTASSOC PACKAGE (CONCAT (OR (CAR (CL:PACKAGE-NICKNAMES PACKAGE))
                                           (CL:PACKAGE-NAME PACKAGE))
                                       ":")
                   *WHO-LINE-PACKAGE-NAME-CACHE*])

(SET-PACKAGE-INTERACTIVELY
  [LAMBDA NIL                                                    ; Edited 15-Mar-2024 07:17 by lmm
                                                                 ; Edited 12-Apr-2023 17:44 by lmm
                                                                 ; Edited 18-Mar-87 13:13 by smL
    ;;
    ;; Let the user interactivly change the current package
    ;;
    (LET* [PKG (MAIN (FOR PN IN '("INTERLISP" "XCL-USER" "USER") WHEN (SETQ PKG (CL:FIND-PACKAGE PN))
                     COLLECT (LIST PKG PN)))
           (SYSPKG (FOR PN
                      IN '("LISP" "XEROX-COMMON-LISP" "D-ASSEM" "FASL" "KEYWORD" "CASH-FILE" "SEDIT"
                                "SYSTEM" "COMPILER" "HASH-FILE" "CONDITIONS" "DEBUGGER" "LOOP")
                      WHEN (SETQ PKG (CL:FIND-PACKAGE PN)) COLLECT (LIST PKG PN)))
           (BOTH (APPEND MAIN SYSPKG))
           [UNSORTED (FOR PKG IN (CL:LIST-ALL-PACKAGES) WHEN (NOT (ASSOC PKG BOTH))
                        COLLECT (LIST PKG (OR (CAR (CL:PACKAGE-NICKNAMES PKG))
                                              (CL:PACKAGE-NAME PKG]
           [USERS (SORT UNSORTED (FUNCTION (LAMBDA (X Y)
                                             (ALPHORDER (CADR X)
                                                        (CADR Y]
           [ITEMS (FOR X IN (APPEND MAIN USERS SYSPKG) COLLECT (LIST (CADR X)
                                                                     (CAR X]
           (SELECTION (MENU (create MENU
                                    TITLE _ "Select package"
                                    ITEMS _ ITEMS
                                    CENTERFLG _ T]
          (IF SELECTION
              THEN (IF (SHIFTDOWNP 'SHIFT)
                       THEN (WHO-LINE-COPY-INSERT (CONCAT (CL:PACKAGE-NAME SELECTION)
                                                          ":"))
                   ELSE (CL:IN-PACKAGE (CL:PACKAGE-NAME SELECTION])

(SET-TTY-PACKAGE-INTERACTIVELY
  [LAMBDA NIL                                                    (* smL "28-Oct-86 09:49")

;;;

;;; Interactivly let the user change the package of the current TTY process

;;;

    (PROCESS.EVAL (TTY.PROCESS)
          '(SET-PACKAGE-INTERACTIVELY)
          T])

)


(DEFGLOBALVAR *WHO-LINE-PACKAGE-NAME-CACHE* (LIST NIL)
                                            "An AList used to cache package names, together with their
                                            terminating ':'s")


(CL:DEFPARAMETER *WHO-LINE-PACKAGE-ENTRY* '("Pkg" (CURRENT-TTY-PACKAGE)
                                                  10 SET-TTY-PACKAGE-INTERACTIVELY (SETQ
                                                                                    *WHO-LINE-PACKAGE-NAME-CACHE*
                                                                                    (LIST NIL))
                                                  "Package of the current TTY process")
                                           "Who-Line entry for displaying the package of the current TTY
```

```
                                                process")

;;
;; Showing / changing the current tty process readtable

(DEFINEQ

(CURRENT-TTY-READTABLE-NAME
  [LAMBDA NIL                                                    (* smL "28-Oct-86 19:13")

;;;
;;; Return the name of the readtable of the current TTY process

;;;

    (OR (READTABLEPROP (PROCESS.EVALV (TTY.PROCESS)
                               '*READTABLE*)
              'NAME)
        "Unknown"])

(SET-READTABLE-INTERACTIVELY
  [LAMBDA NIL                                                    (* smL "10-Nov-86 18:36")
    ;;
    ;; Let the user interactivly change the current readtable
    ;;
    (DECLARE (GLOBALVARS \READTABLEHASH))
    (LET [(READTABLE (MENU (create MENU
                                   TITLE _ "Select readtable"
                                   ITEMS _ [LET ((READTABLES NIL))
                                                [MAPHASH \READTABLEHASH (FUNCTION (LAMBDA (VALUE NAME)
                                                                                    (push READTABLES
                                                                                          (LIST NAME VALUE]
                                                (SORT READTABLES (FUNCTION (LAMBDA (X Y)
                                                                             (ALPHORDER (CAR X)
                                                                                    (CAR Y]
                                   CENTERFLG _ T]
          (if (READTABLEP READTABLE)
              then (SETQ *READTABLE* READTABLE])

(SET-TTY-READTABLE-INTERACTIVELY
  [LAMBDA NIL                                                    (* smL "28-Oct-86 09:51")

;;;
;;; Interactivly let the user change the package of the current TTY readtable

;;;

    (PROCESS.EVAL (TTY.PROCESS)
           '(SET-READTABLE-INTERACTIVELY)
           T])

)

(CL:DEFPARAMETER *WHO-LINE-READTABLE-ENTRY* '("Rdtbl" (CURRENT-TTY-READTABLE-NAME)
                                                      10 SET-TTY-READTABLE-INTERACTIVELY NIL "Readtable of
                                                      the current TTY process")
                                                    "Who-Line entry for displaying the name of the ReadTable of
                                                    the current TTY process")

;;
;; Showing / changing the current tty process

(DEFINEQ

(WHO-LINE-TTY-PROCESS
  [LAMBDA NIL                                                    (* smL "28-Oct-86 09:54")

;;;
;;; Return the name of the current TTY process

;;;

    (PROCESSPROP (TTY.PROCESS)
           'NAME])

(CHANGE-TTY-PROCESS-INTERACTIVELY
```

```
  [LAMBDA NIL                                                    (* smL "10-Nov-86 18:36")
    (DECLARE (GLOBALVARS \PROCESSES))
    (LET [(NEW-PROC (MENU (create MENU
                                  TITLE _ "Give TTY to process"
                                  CENTERFLG _ T
                                  ITEMS _ (SORT (for PROC in \PROCESSES collect (LIST (PROCESSPROP PROC 'NAME)
                                                                                      PROC))
                                                (FUNCTION (LAMBDA (X Y)
                                                            (ALPHORDER (CAR X)
                                                                   (CAR Y]
          (if NEW-PROC
              then (TTY.PROCESS NEW-PROC])

)


(CL:DEFPARAMETER *WHO-LINE-TTY-PROC-ENTRY* '("Tty" (WHO-LINE-TTY-PROCESS)
                                                   15 CHANGE-TTY-PROCESS-INTERACTIVELY NIL "The current
                                                   TTY process")
                                           "Who-Line entry for displaying the name of the current TTY
                                           process")

;;
;; Showing / changing the currently connected directory

(DEFINEQ

(WHO-LINE-CURRENT-DIRECTORY
  [LAMBDA NIL                                                    ; Edited  3-Feb-89 14:52 by smL

;;; Get the currently connected directory
      ;; First, update the cached directory / namestring pair to reflect the current TTY proc
      (DECLARE (GLOBALVARS *WHO-LINE-LAST-DIRECTORY*))
      ;; The connected directory is looked up in the TTY process, in case one day it becomes a per-process var
      [LET [(CONNECTED-DIRECTORY (PROCESS.EVALV (TTY.PROCESS)
                                       '*DEFAULT-PATHNAME-DEFAULTS*]
                                                                 ; The CAR contains the path, the CDR contains a string version
                                                                 ; of the path
           (if (NEQ CONNECTED-DIRECTORY (CAR *WHO-LINE-LAST-DIRECTORY*))
               then                                              ; The connected directory has changed
                    (change (CAR *WHO-LINE-LAST-DIRECTORY*)
                            CONNECTED-DIRECTORY)                 ; Put the host name last, since that is least important
                    (change (CDR *WHO-LINE-LAST-DIRECTORY*)
                            (if (CL:PATHNAME-DIRECTORY CONNECTED-DIRECTORY)
                                then (CONCAT (CL:PATHNAME-DIRECTORY CONNECTED-DIRECTORY)
                                             " on {"
                                             (CL:PATHNAME-HOST CONNECTED-DIRECTORY)
                                             "}")
                              else (CONCAT "{" (CL:PATHNAME-HOST CONNECTED-DIRECTORY)
                                           "}")))                ; Update the list of known directories
                    (LET ((DIR-NAME (CL:NAMESTRING CONNECTED-DIRECTORY)))
                         (if (NOT (CL:MEMBER DIR-NAME *WHO-LINE-DIRECTORIES* :TEST #'STRING-EQUAL))
                             then (MERGEINSERT DIR-NAME (SORT *WHO-LINE-DIRECTORIES* #'UALPHORDER]
      ;; Return the namestring of the current dir
      (CDR *WHO-LINE-LAST-DIRECTORY*])

(SET-CONNECTED-DIRECTORY-INTERACTIVELY
  [LAMBDA NIL                                                    ; Edited 12-Apr-2023 08:00 by lmm
                                                                 ; Edited  9-Jun-87 08:57 by smL

;;; Let the user interactivly change the current connected directory

      (DECLARE (GLOBALVARS *WHO-LINE-DIRECTORIES*))
      ;; If the user selects an item while holding down a shift key, copy-insert the name of the directory instead of connecting to it
      (SETQ *WHO-LINE-DIRECTORIES* (SUBSET *WHO-LINE-DIRECTORIES* (FUNCTION DIRECTORYNAMEP)))
      (IF (SHIFTDOWNP 'SHIFT)
          THEN (LET [(NEW-DIRECTORY (MENU (create MENU
                                                  TITLE _ "Type in directory name:"
                                                  ITEMS _ *WHO-LINE-DIRECTORIES*]
                    (IF NEW-DIRECTORY
                        THEN (WHO-LINE-COPY-INSERT NEW-DIRECTORY)))
        ELSE (LET [(NEW-DIRECTORY (MENU (create MENU
                                                TITLE _ "Connect to:"
                                                ITEMS _ (CONS "* Other *" *WHO-LINE-DIRECTORIES*]
                  (if NEW-DIRECTORY
                      then (if (STRING-EQUAL NEW-DIRECTORY "* Other *")
                               then (CLEARW PROMPTWINDOW)
                                    (SETQ NEW-DIRECTORY (PROMPTFORWORD "Connect to directory "
                                                                       (CL:NAMESTRING (PROCESS.EVALV (TTY.PROCESS)
                                                                                                     '*DEFAULT-PATHNAME-DEFAULTS*))
                                                                       NIL PROMPTWINDOW NIL 'TTY NIL)))
```

```
                        (if NEW-DIRECTORY
                            then (ALLOW.BUTTON.EVENTS)          ; Should do this in the current TTY process, in case the
                                                                ; conntected directory is a per-process var
                                 (CNDIR NEW-DIRECTORY])

)

(DEFGLOBALVAR *WHO-LINE-DIRECTORIES* `(,LOGINHOST/DIR)
                                    "Cached list of known directories for the Who-Line Directory entry")

(DEFGLOBALVAR *WHO-LINE-LAST-DIRECTORY* (LET ((NAMESTRING (CL:NAMESTRING *DEFAULT-PATHNAME-DEFAULTS*)))
                                          (CONS (PATHNAME NAMESTRING)
                                                (MKSTRING NAMESTRING)))
                                       "Cached name of the current connected directory for the Who-Line
                                       Directory entry")

(CL:DEFPARAMETER *WHO-LINE-DIRECTORY-ENTRY* '("Dir" (WHO-LINE-CURRENT-DIRECTORY)
                                                    30 SET-CONNECTED-DIRECTORY-INTERACTIVELY
                                                    (SETQ *WHO-LINE-LAST-DIRECTORY* (CONS NIL NIL))
                                                    "The currently connected directory")
                                              "Who-Line entry for displaying the name of the currently
                                              connected directory")

;;
;; Showing / changing the current VMem utilization

(DEFINEQ

(WHO-LINE-VMEM
  [LAMBDA NIL                                                   ; Edited 16-Jun-94 21:12 by kaplan

;;;
;;; Compute the percentage of vmem in use.
;;;
    (DECLARE (GLOBALVARS *WHO-LINE-LAST-VMEM* \LASTVMEMFILEPAGE \InterfacePage \IFPValidKey))
    ;;
    ;; Compute the percentage of vmem in use.  The ratio is the amount in use (computed by (VMEMSIZE)) divided by the amount available (stored in
    ;; \LASTVMEMFILEPAGE).  We multiply by 100 to get a percentage, round to an integer, and do it all in such a way as to ensure we don't cons any
    ;; FIXPs.
    ;; The basic code here is due to Mike Dixon.
    ;;
    (LET* ((ONE-PERCENT-VMEM (IQUOTIENT (IPLUS (MAX \LASTVMEMFILEPAGE (TIMES 2 65535))
                                               50)
                                        100))
           (VMEM-PERCENT (IQUOTIENT (IPLUS (VMEMSIZE)
                                           (RSH ONE-PERCENT-VMEM 1))
                                    ONE-PERCENT-VMEM))
           (VMEM-CONSISTENT? (.VMEM.CONSISTENTP.)))
          ;;
          ;; We cache the last VMem info and the string-translation of it in the var *WHO-LINE-LAST-VMEM*.  That way, we don't have to alloc a
          ;; new string all the time.  We do, however, have to make sure the cached info in correct.
          ;;
          (if [NOT (AND (EQ VMEM-CONSISTENT? (CADR *WHO-LINE-LAST-VMEM*))
                        (EQP VMEM-PERCENT (CAR *WHO-LINE-LAST-VMEM*]
              then (change (CAR *WHO-LINE-LAST-VMEM*)
                           VMEM-PERCENT)
                   (change (CADR *WHO-LINE-LAST-VMEM*)
                           VMEM-CONSISTENT?)
                   (change (CADDR *WHO-LINE-LAST-VMEM*)
                           (CONCAT (if VMEM-CONSISTENT?
                                       then " "
                                     else "*")
                                   VMEM-PERCENT "%%")))
          ;;
          ;; Return the info string
          ;;
          (CADDR *WHO-LINE-LAST-VMEM*])

(WHO-LINE-SAVE-VMEM
  [LAMBDA NIL                                                   (* smL "29-Oct-86 11:22")

;;;
```

```
;;; Save the VMem, if the user really wants to

;;;

    (if [MENU (create MENU
                      TITLE _ "Save VMem?"
                      CENTERFLG _ T
                      ITEMS _ '(("Yes" T)
                                ("No" NIL]
        then (SAVEVM])

)


(DEFGLOBALVAR *WHO-LINE-LAST-VMEM* (LIST 0 NIL NIL)
                                   "Cached value for storing the last VMem information for the Who-Line
                                   VMem entry")


(CL:DEFPARAMETER *WHO-LINE-VMEM-ENTRY* '("VMem" (WHO-LINE-VMEM)
                                                 5 WHO-LINE-SAVE-VMEM (SETQ *WHO-LINE-LAST-VMEM*
                                                                            (LIST 0 NIL NIL))
                                                 "Percentage of VMem currently in use")
                                               "Who-Line entry for displaying the current VMem utilization")

(DECLARE%: EVAL@COMPILE DONTCOPY DONTEVAL@LOAD

(FILESLOAD (LOADCOMP)
       LLFAULT MODARITH)

(CHECKIMPORTS '(LLPARAMS)
       T)
)

;;
;; Showing the percent of symbol-space currently used


(CL:DEFUN WHO-LINE-SYMBOL-SPACE ()
   "Return a string describing the percentage of symbol space in use"
   (LET ((TOTAL-SYMBOL-SPACE (UNFOLD (CL:1+ \LastAtomPage)
                                     WORDSPERCELL))
         (SYMBOL-SPACE-IN-USE (FOLDHI \AtomFrLst CELLSPERPAGE)))
        ;; Only recompute the display string when the fraction of space has changed.  This saves us the effort of CONSing up the string each time.
        (CL:UNLESS (AND (EQL (CL:FIRST *WHO-LINE-SYMBOL-SPACE*)
                             TOTAL-SYMBOL-SPACE)
                        (EQL (CL:SECOND *WHO-LINE-SYMBOL-SPACE*)
                             SYMBOL-SPACE-IN-USE))
            [CL:SETF (CL:FIRST *WHO-LINE-SYMBOL-SPACE*)
                   TOTAL-SYMBOL-SPACE
                   (CL:SECOND *WHO-LINE-SYMBOL-SPACE*)
                   SYMBOL-SPACE-IN-USE
                   (CL:THIRD *WHO-LINE-SYMBOL-SPACE*)
                   (CL:FORMAT NIL "~3D%%" (- 100 (ROUND (- 100 (/ (CL:* SYMBOL-SPACE-IN-USE 100)
                                                                  TOTAL-SYMBOL-SPACE])
        (CL:THIRD *WHO-LINE-SYMBOL-SPACE*)))


(DEFGLOBALVAR *WHO-LINE-SYMBOL-SPACE* (LIST NIL NIL NIL "Remembers the previous who-line symbol space"))


(CL:DEFPARAMETER *WHO-LINE-SYMBOL-SPACE-ENTRY* '("Syms" (WHO-LINE-SYMBOL-SPACE)
                                                        4 NIL (SETQ *WHO-LINE-SYMBOL-SPACE*
                                                                    (LIST NIL NIL NIL))
                                                        "Percentage of symbol space currently in use")
                                                      "Who-line entry for displaying percent of symbol space in
                                                      use")

;;
;; Showing the current time

(DEFINEQ

(WHO-LINE-TIME
  [LAMBDA NIL                                                    ; Edited 14-Jan-87 12:48 by smL

;;;

;;; Return the current time as a string.  Avoid CONSing as much as possible.

;;;

    (DECLARE (GLOBALVARS *WHO-LINE-TIMER* *WHO-LINE-OLD-TIME*))
    (if (TIMEREXPIRED? *WHO-LINE-TIMER* 'SECONDS)
```

```
        then ;;
             ;; Reset the timer, and return the new time
             ;;
             (LET ((NOW (IDATE)))
                  (SETQ *WHO-LINE-TIMER* (SETUPTIMER (DIFFERENCE 60 (REMAINDER NOW 60))
                                                     (CONSTANT (SETUPTIMER 0 NIL 'SECONDS))
                                                     'SECONDS))
                  (SETQ *WHO-LINE-OLD-TIME* (GDATE NOW (CONSTANT (DATEFORMAT NO.SECONDS))
                                                   *WHO-LINE-OLD-TIME*))
                  *WHO-LINE-OLD-TIME*)
       else ;;
             ;; The timer hasn't expired, so the old time is good enough
             ;;
             *WHO-LINE-OLD-TIME*])
```

(WHO-LINE-SET-TIME

```
  [LAMBDA NIL                                                    ; Edited 17-Mar-87 18:20 by smL
```

;;;

;;; Set the time from the network, if the user really wants to

;;;

```
    (COND
       ((SHIFTDOWNP 'SHIFT)
        ;; Selection with a shift key down causes the current time to be bksysbuf'ed
        (WHO-LINE-COPY-INSERT *WHO-LINE-OLD-TIME*))
       ([MENU (create MENU
                      TITLE _ "Set time?"
                      CENTERFLG _ T
                      ITEMS _ '(("Yes" T)
                                ("No" NIL]
        ;; The user wants to reset the time
        (SETTIME])

)


(DEFGLOBALVAR *WHO-LINE-TIMER* (SETUPTIMER (DIFFERENCE 60 (REMAINDER (IDATE)
                                                                     60))
                                           NIL
                                           'SECONDS)
                               "Timer for controlling updates of the Who-Line Time entry")


(DEFGLOBALVAR *WHO-LINE-OLD-TIME* (DATE (DATEFORMAT NO.SECONDS))
                                  "Cached value for the Who-Line Time entry")


(CL:DEFPARAMETER *WHO-LINE-TIME-ENTRY*
   '("Time" (WHO-LINE-TIME)
            17 WHO-LINE-SET-TIME [PROGN (SETQ *WHO-LINE-OLD-TIME* (DATE (DATEFORMAT NO.SECONDS)))
                                        (SETQ *WHO-LINE-TIMER* (SETUPTIMER (DIFFERENCE 60 (REMAINDER (IDATE)
                                                                                                     60))
                                                                           NIL
                                                                           'SECONDS]
            "Time of day")
   "Who-Line entry for displaying the current time of day")

(DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE

(APPENDTOVAR \SYSTEMTIMERVARS (*WHO-LINE-TIMER* SECONDS))
)


;;; -------------------------------------------------------------------

;;; Some as yet un-debugged entries.  Try at your own risk.
;;
;; Showing the machine-active entry

(DEFINEQ
```

(WHO-LINE-SHOW-ACTIVE

```
  [LAMBDA NIL                                                    ; Edited 20-Apr-87 09:58 by smL
```

;;; Update the who-line active indicator, if it is time

```
    (DECLARE (GLOBALVARS *WHO-LINE* *WHO-LINE-ACTIVE-TIMER* *WHO-LINE-ACTIVE-PERIOD*))
    [if (TIMEREXPIRED? *WHO-LINE-ACTIVE-TIMER* 'MILLISECONDS)
        then ;; A second has passed, so update the indicator if we can
             (\UPDATE-WHO-LINE-ACTIVE-FLAG *WHO-LINE*)
             ;; Reset the timer
             (SETQ *WHO-LINE-ACTIVE-TIMER* (SETUPTIMER *WHO-LINE-ACTIVE-PERIOD* *WHO-LINE-ACTIVE-TIMER*
                                                       'MILLISECONDS]
    ;; Always return the same thing
    " "])
```

(\UPDATE-WHO-LINE-ACTIVE-FLAG

```
  [LAMBDA (WINDOW)                                          ; Edited 20-Apr-87 09:58 by smL
```

;;; Flip the active-indicator in the who-line

```
    (for ENTRY in (WINDOWPROP WINDOW 'ENTRIES) thereis [with WHO-LINE-ENTRY ENTRY (AND (LISTP FORM)
                                                                                       (EQ (CAR FORM)
                                                                                           'WHO-LINE-SHOW-ACTIVE]
       finally (if $$VAL
                   then (with WHO-LINE-ENTRY $$VAL (BLTSHADE BLACKSHADE WINDOW VALUE-START 2 (DIFFERENCE VALUE-END
                                                                                                 VALUE-START)
                                                              (DIFFERENCE (WINDOWPROP WINDOW 'HEIGHT)
                                                                          4)
                                                              'INVERT])
```

(\PERIODICALLY-WHO-LINE-SHOW-ACTIVE

```
  [LAMBDA NIL                                               ; Edited 14-Jan-87 12:50 by smL
```

;;;

;;; Update the who-line active indicator, if it is time

;;; This is designed to be run on the \PERIODIC.INTERRUPT hook.

;;;

```
    (DECLARE (GLOBALVARS *WHO-LINE-ACTIVE-TIMER* *WHO-LINE* *WHO-LINE-ACTIVE-PERIOD*))
    (if (TIMEREXPIRED? *WHO-LINE-ACTIVE-TIMER* 'MILLISECONDS)
        then ;;
             ;; A second has passed, so update the indicator if we can
             ;;
             ;; But only if the who-line is on the top
             ;;
             (if (AND (OPENWP *WHO-LINE*)
                      (TOPWP *WHO-LINE*))
                 then ;;
                      ;; The who-line is on the top, so we can update it
                      ;;
                      (\UPDATE-WHO-LINE-ACTIVE-FLAG *WHO-LINE*))
             ;;
             ;; Reset the timer
             ;;
             (SETQ *WHO-LINE-ACTIVE-TIMER* (SETUPTIMER *WHO-LINE-ACTIVE-PERIOD* *WHO-LINE-ACTIVE-TIMER*
                                                       'MILLISECONDS])

)

(DEFGLOBALVAR *WHO-LINE-ACTIVE-PERIOD* 500
   "Interval between updating the Who-Line activity entry")

(DEFGLOBALVAR *WHO-LINE-ACTIVE-TIMER* (SETUPTIMER *WHO-LINE-ACTIVE-PERIOD* NIL 'MILLISECONDS)
                                      "Timer for controlling updating of the Who-Line activity entry")

(CL:DEFPARAMETER *WHO-LINE-SHOW-ACTIVE-ENTRY* '("" (WHO-LINE-SHOW-ACTIVE)
                                                   2 NIL (SETQ *WHO-LINE-ACTIVE-TIMER*
                                                               (SETUPTIMER *WHO-LINE-ACTIVE-PERIOD* NIL
                                                                           'MILLISECONDS))
                                                   "Indication of machine activity")
                                                 "Who-Line entry for displaying the activity of the
                                                 machine")

(DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE
```

```
(APPENDTOVAR \SYSTEMTIMERVARS (*WHO-LINE-ACTIVE-TIMER* MILLISECONDS))
)

;;
;; Showing / changing the current reader profile

(DEFINEQ

(CURRENT-PROFILE
  [LAMBDA NIL                                                    ; Edited 12-Jan-87 14:36 by smL

;;;

;;; Return the name of the current reader profile of the current TTY process

;;;

    (XCL:PROFILE-NAME (PROCESS.EVALV (TTY.PROCESS)
                              'XCL:*PROFILE*])

(SET-PROFILE-INTERACTIVELY
  [LAMBDA NIL                                                    ; Edited 12-Jan-87 14:33 by smL
    ;;
    ;; Let the user interactivly change the current reader profile
    ;;
    (LET [(PROFILE (MENU (create MENU
                                 TITLE _ "Select profile"
                                 ITEMS _ (SORT (for PROFILE in (XCL:LIST-ALL-PROFILES) bind PROFILE-NAME
                                                  collect (XCL:PROFILE-NAME PROFILE)))
                                 CENTERFLG _ T]
         (if PROFILE
             then (XCL:RESTORE-PROFILE PROFILE])

(SET-TTY-PROFILE-INTERACTIVELY
  [LAMBDA NIL                                                    ; Edited 12-Jan-87 14:33 by smL

;;;

;;; Interactivly let the user change the reader profile of the current TTY process

;;;

    (PROCESS.EVAL (TTY.PROCESS)
           '(SET-PROFILE-INTERACTIVELY)
           T])

)


(CL:DEFPARAMETER *WHO-LINE-PROFILE-ENTRY* '("Profile" (CURRENT-PROFILE)
                                                  10 SET-TTY-PROFILE-INTERACTIVELY NIL "The read/write
                                                  profile of the current TTY process")
                                          "Who-Line entry for displaying the current read/write profile")

;;
;; Showing the state of the current TTY process

(DEFINEQ

(WHO-LINE-TTY-STATE
  [LAMBDA NIL                                                    ; Edited 17-Apr-87 18:26 by smL

;;; Find out what state the current TTY process is in

    (LET ((PROC (TTY.PROCESS)))
         (COND
            ((NULL PROC)
             ;; No tty process?  Never happens now, but maybe allowed in future.
             "")
            ((EQ PROC (THIS.PROCESS))
             ;; Check explicitly for us being tty, since in that case PROC is not a valid stack pointer (we're running).
             "Who-Line")
            ((PROCESS.EVALV PROC '*WHO-LINE-STATE*))
            ((NOT (PROCESS.FINISHEDP PROC))
             (for I from 0 by -1 bind FRAMENAME while (SETQ FRAMENAME (STKNTHNAME I PROC))
                unless (MEMB FRAMENAME *WHO-LINE-STATE-UNINTERESTING-FNS*)
                do ;; Walk back process looking for interesting frame name.  This search is non-linear in that each iteration takes a little longer,
                   ;; but we expect it to terminate early.
```

```
                    (RETURN (OR (GETPROP FRAMENAME 'WHO-LINE-STATE)
                                FRAMENAME])
```

(WHO-LINE-WHAT-IS-RUNNING

```
  [LAMBDA NIL                                                    ; Edited 14-Jan-87 12:51 by smL
```

;;;

;;; When run under a (PROCESS.EVAL <proc> '(WHO-LINE-WHAT-IS-RUNNING) T), returns the name of the current running frame in the process

;;;

```
    (DECLARE (GLOBALVARS *WHO-LINE-STATE-UNINTERESTING-FNS*))
    (PROG ((POS-HOLDER (CONSTANT (LIST NIL)))
           POS)
     ;;
     ;; We use the POS-HOLDER to hold an old stack pointer, so that we don't have to alloc one
     ;;
          (SETQ POS (STKPOS '\PROCESS.EVAL1 NIL NIL (CAR POS-HOLDER)))
          (COND
             (POS (change (CAR POS-HOLDER)
                          POS))
             (T (RETURN NIL)))
      LP  (SETQ POS (STKNTH 1 POS POS))
          (COND
             ((NULL POS)
              (RETURN NIL))
             ((MEMB (STKNAME POS)
                    *WHO-LINE-STATE-UNINTERESTING-FNS*)          ; Ignore any uninteresting fns
              (GO LP))
             (T (RETURN (PROG1 (STKNAME POS)
                               (RELSTK POS])

)

(CL:DEFVAR *WHO-LINE-STATE* NIL
   "Cached state shown in the Who-Line State entry")

(DEFGLOBALVAR *WHO-LINE-STATE-UNINTERESTING-FNS* '(BLOCK ERRORSET OBTAIN.MONITORLOCK MONITOR.AWAIT.EVENT
                                                        AWAIT.EVENT SI::*UNWIND-PROTECT*)
                                                "Uninteresting fns to skip over in the Who-Line State
                                                entry")

(CL:DEFPARAMETER *WHO-LINE-TTY-STATE-ENTRY* '("State" (WHO-LINE-TTY-STATE)
                                                       15 NIL (SETQ *WHO-LINE-STATE* NIL)
                                                       "Running state of the current TTY process")
                                            "Who-Line entry for showing the running state of the current
                                            TTY process")

(PUTPROPS AWAIT.EVENT WHO-LINE-STATE "Block")

(PUTPROPS BLOCK WHO-LINE-STATE "Block")

(PUTPROPS EXCHANGEPUPS WHO-LINE-STATE "Net I/O")

(PUTPROPS GETPUP WHO-LINE-STATE "Net I/O")

(PUTPROPS SENDPUP WHO-LINE-STATE "Net I/O")

(PUTPROPS WAIT.FOR.TTY WHO-LINE-STATE "TTY wait")

(PUTPROPS \TTYBACKGROUND WHO-LINE-STATE "TTY wait")

(PUTPROPS \WAITFORSYSBUFP WHO-LINE-STATE "TTY wait")

(PUTPROPS \\getkey WHO-LINE-STATE "TTY wait")

(PUTPROPS \SENDLEAF WHO-LINE-STATE "Net I/O")

(PUTPROPS PUTSEQUIN WHO-LINE-STATE "Net I/O")

(PUTPROPS \LEAF.READPAGES WHO-LINE-STATE "Net I/O")
```

;;; --

;;; Default options for the who-line

```
(DEFGLOBALVAR *WHO-LINE-ENTRIES*
```

```
   `(,*WHO-LINE-PACKAGE-ENTRY* ,*WHO-LINE-READTABLE-ENTRY* ,*WHO-LINE-TTY-PROC-ENTRY*
          ,*WHO-LINE-DIRECTORY-ENTRY* ,*WHO-LINE-VMEM-ENTRY* ,*WHO-LINE-TIME-ENTRY*)
   "List of all the entries to show in the Who-Line")


(DEFGLOBALVAR *WHO-LINE-ENTRY-REGISTRY* (LIST *WHO-LINE-USER-ENTRY* *WHO-LINE-HOST-NAME-ENTRY*
                                              *WHO-LINE-PACKAGE-ENTRY* *WHO-LINE-READTABLE-ENTRY*
                                              *WHO-LINE-TTY-PROC-ENTRY* *WHO-LINE-DIRECTORY-ENTRY*
                                              *WHO-LINE-VMEM-ENTRY* *WHO-LINE-SYMBOL-SPACE-ENTRY*
                                              *WHO-LINE-TIME-ENTRY* *WHO-LINE-SHOW-ACTIVE-ENTRY*
                                              *WHO-LINE-PROFILE-ENTRY* *WHO-LINE-TTY-STATE-ENTRY*)
                                        "List of all known Who-Line entries.")


(DEFGLOBALVAR *WHO-LINE-ANCHOR* '(:CENTER :TOP)
                                      "Location to place the Who-Line")


(DEFGLOBALVAR *WHO-LINE-NAME-FONT* (FONTCREATE '(HELVETICA 8 BOLD))
                                     "Font to use to show entry labels in the Who-Line")


(DEFGLOBALVAR *WHO-LINE-VALUE-FONT* (FONTCREATE '(GACHA 8))
                                      "Font to use to show the entry values in the Who-Line")


(DEFGLOBALVAR *WHO-LINE-DISPLAY-NAMES?* T
   "Flag for enabling or disabling the display of entry names in the Who-Line")


(DEFGLOBALVAR *WHO-LINE-COLOR* :WHITE "Color of the Who-Line -- one of :WHITE or :BLACK")


(DEFGLOBALVAR *WHO-LINE-TITLE* NIL
   "The window title of the Who-Line")


(DEFGLOBALVAR *WHO-LINE-BORDER* 2
   "The border width of the Who-Line window")


(DEFGLOBALVAR *WHO-LINE-UPDATE-INTERVAL* 100
   "Update interval for the Who-Line, in milliseconds")


;;; -------------------------------------------------------------------------

;;; Internal fns

(DEFINEQ

(REDISPLAY-WHO-LINE
  [LAMBDA (WINDOW)                                          ; Edited 17-Apr-87 19:06 by smL

;;; Redisplay the entire who-line, including the names of the fields

    (WITH-WHO-LINE WINDOW
           ;;
           ;; Set the display characteristics of the window, according to its color
           (DSPSOURCETYPE [SELECTQ (WINDOWPROP WINDOW 'COLOR)
                                (:WHITE 'INPUT)
                                (:BLACK 'INVERT)
                                (ERROR "Illegal color for Who-Line" (WINDOWPROP WINDOW 'COLOR]
                  WINDOW)
           (DSPTEXTURE [SELECTQ (WINDOWPROP WINDOW 'COLOR)
                           (:WHITE WHITESHADE)
                           (:BLACK BLACKSHADE)
                           (ERROR "Illegal color for Who-Line" (WINDOWPROP WINDOW 'COLOR]
                  WINDOW)
           ;;
           ;; Clear the window
           (CLEARW WINDOW)
           (for ITEM in (WINDOWPROP WINDOW 'ENTRIES) do (replace (WHO-LINE-ENTRY INVERTED?) of ITEM with NIL))
           ;;
           ;; Display the labels if we should
           (if (WINDOWPROP WINDOW 'DISPLAY-NAMES?)
               then (DSPFONT (WINDOWPROP WINDOW 'NAME-FONT)
                             WINDOW)
                    (for ITEM in (WINDOWPROP WINDOW 'ENTRIES) bind (FONT _ (WINDOWPROP WINDOW 'NAME-FONT))
                       do (MOVETO (fetch (WHO-LINE-ENTRY NAME-START) of ITEM)
                                  (PLUS (FONTPROP FONT 'DESCENT)
                                        (QUOTIENT (DIFFERENCE (WINDOWPROP *WHO-LINE* 'HEIGHT)
                                                              (FONTPROP FONT 'HEIGHT))
```

```
                                        2))
                                WINDOW)
                        (PRIN1 (fetch (WHO-LINE-ENTRY NAME) of ITEM)
                                WINDOW)))
        ;;
        ;; Display the values
        (DSPFONT (WINDOWPROP WINDOW 'VALUE-FONT)
                WINDOW)
        (UPDATE-WHO-LINE WINDOW (WINDOWPROP WINDOW 'ENTRIES)
                T)
        ;;
        ;; Reset the timer for the next update
        (SETQ *WHO-LINE-UPDATE-TIMER* (SETUP-WHOLINE-TIMER *WHO-LINE-UPDATE-TIMER*])
```

(PERIODICALLY-UPDATE-WHO-LINE

```
  [LAMBDA NIL                                                    ; Edited 27-Jan-88 10:11 by smL
```

;;;

;;; Update the current who-line window every so often. This is designed to be placed on the list of BACKBROUNDFNS.

;;;

```
    (DECLARE (GLOBALVARS \IDLING))
    (CL:WHEN (TIMEREXPIRED? *WHO-LINE-UPDATE-TIMER* 'TICKS)
        (CL:WHEN (AND (BOUNDP '*WHO-LINE*)
                      (NOT \IDLING))                             ; Don't bother to wait and update if the window is owned by
                                                                 ; someone.
            [WITH-AVAILABLE-WHO-LINE *WHO-LINE* (if (AND (OPENWP *WHO-LINE*)
                                                         (GETWINDOWPROP *WHO-LINE* 'VALID))
                                                    then (UPDATE-WHO-LINE *WHO-LINE* (GETWINDOWPROP
                                                                                       *WHO-LINE*
                                                                                       'ENTRIES])
        (SETQ *WHO-LINE-UPDATE-TIMER* (SETUP-WHOLINE-TIMER *WHO-LINE-UPDATE-TIMER*)))])
```

(SETUP-WHOLINE-TIMER

```
  [LAMBDA (OLD-TIMER)                                            ; Edited 18-Mar-87 11:14 by smL
    (SETUPTIMER (WINDOWPROP *WHO-LINE* 'UPDATE-INTERVAL)
           OLD-TIMER
           'TICKS])
```

(UPDATE-WHO-LINE

```
  [LAMBDA (WINDOW WHO-LINE-ENTRIES ALWAYS?)                      ; Edited 17-Apr-87 19:05 by smL
```

;;; Update the window to show the current who-line stats

```
    (WITH-WHO-LINE WINDOW
           ;;
           ;; Update all the entries that have changed
           (for ENTRY in WHO-LINE-ENTRIES bind (VALUE-BOTTOM _ (GETWINDOWPROP WINDOW 'VALUE-BOTTOM))
                                               (STREAM _ (GETWINDOWPROP WINDOW 'TEMP-STREAM))
                                               (HEIGHT _ (GETWINDOWPROP WINDOW 'HEIGHT))
                                               (BLACK-WINDOW-P _ (EQ (WINDOWPROP WINDOW 'COLOR)
                                                                     :BLACK))
              do (with WHO-LINE-ENTRY ENTRY                      ; If the node is inverted, the user is mousing it, so don't update it
                       (if (NOT INVERTED?)
                           then (if ALWAYS?
                                    then (EVAL RESET-FORM))
                                (LET ((VALUE (EVAL FORM)))
                                     ;;
                                     ;; Only update if the value has changed, or we are ordered to.
                                     (if (OR ALWAYS? (NOT (EQUAL VALUE PREV-VALUE)))
                                         then ;;
                                              ;; Print the new value
                                              (MOVETO VALUE-START VALUE-BOTTOM STREAM)
                                              (BLTSHADE BLACKSHADE STREAM VALUE-START 0 (DIFFERENCE VALUE-END
                                                                                                   VALUE-START)
                                                     HEIGHT
                                                     'ERASE)
                                              (DSPFONT (DSPFONT NIL WINDOW)
                                                     STREAM)
                                              (PRIN1 VALUE STREAM)
                                              (if BLACK-WINDOW-P
                                                  then (BLTSHADE BLACKSHADE STREAM VALUE-START 0 (DIFFERENCE
                                                                                                   VALUE-END

                                                                                                   VALUE-START
```

```
                                                                                    )
                                                    HEIGHT
                                                    'INVERT))
                                        (BITBLT STREAM VALUE-START 0 WINDOW VALUE-START 0 (DIFFERENCE
                                                                                            VALUE-END
                                                                                            VALUE-START
                                                                                            )
                                               HEIGHT
                                               'PAINT)
                                        ;;
                                        ;; Save the value.
                                        ;; We are worried that a form may be re-using a value (to minimize CONS-ing), so we store a
                                        ;; copy of the value rather than the real value.
                                        (SETQ PREV-VALUE (COPYALL VALUE])
```

(WHEN-WHO-LINE-SELECTED-FN

```
[LAMBDA (WINDOW)                                                    ; Edited 27-Jan-88 09:54 by smL

;;;

;;; The button has gone down in the who-line window.

;;; If the control or edit key is down, allow the user to change the entries in the who-line.

;;; If the user selects an item, and it has a when-selected-fn, funcall that fn.

;;;

    (WITH-WHO-LINE WINDOW (TOTOPW WINDOW)
           (GETMOUSESTATE)
           (if (OR (KEYDOWNP 'EDIT)
                   (KEYDOWNP 'CTRL))
               then (WHO-LINE-CONTROL-SELECT)
             else (bind (REGION _ (WINDOWPROP WINDOW 'REGION))
                        (ENTRIES _ (WINDOWPROP WINDOW 'ENTRIES))
                        INVERTED-ITEM CURRENT-ITEM while (MOUSESTATE (NOT UP))
                  do ;;
                     ;; If cursor has left the window, quit tracking
                     ;;
                     (if (NOT (INSIDEP REGION LASTMOUSEX LASTMOUSEY))
                         then (SETQ CURRENT-ITEM NIL)
                              (GO $$OUT))
                     ;;
                     ;; Find out what item we are currently on
                     ;;
                     [SETQ CURRENT-ITEM (for ENTRY in ENTRIES thereis (with WHO-LINE-ENTRY ENTRY
                                                                        (AND (GEQ (LASTMOUSEX WINDOW)
                                                                                  NAME-START)
                                                                             (LEQ (LASTMOUSEX WINDOW)
                                                                                  VALUE-END)
                                                                             (NOT (NULL WHEN-SELECTED-FN]
                     ;;
                     ;; Invert the current choice
                     ;;
                     (if (NEQ INVERTED-ITEM CURRENT-ITEM)
                         then (if INVERTED-ITEM
                                  then (INVERT-WHO-LINE-ENTRY INVERTED-ITEM WINDOW))
                              (if CURRENT-ITEM
                                  then (INVERT-WHO-LINE-ENTRY CURRENT-ITEM WINDOW))
                              (SETQ INVERTED-ITEM CURRENT-ITEM))
                  finally ;;
                          ;; The button went up.  If we were on an item, let it know
                          ;;
                          (if INVERTED-ITEM
                              then (INVERT-WHO-LINE-ENTRY INVERTED-ITEM WINDOW))
                          (if CURRENT-ITEM
                              then (with WHO-LINE-ENTRY CURRENT-ITEM (if WHEN-SELECTED-FN
                                                                         then (APPLY* WHEN-SELECTED-FN)
                                                                              (EVAL RESET-FORM])
```

(WHO-LINE-CONTROL-SELECT

```
[LAMBDA NIL                                                         ; Edited 28-Dec-98 12:56 by rmk:
  "Interactivly let the user add or delete an entry to the WHO-LINE."
  (CL:FLET [(ENTRY-DESCRIPTION (X)
                  (OR (CL:SIXTH X)
```

```
                        (CONCAT "Entry named: " (CL:FIRST X]
            (CASE (MENU (create MENU
                                ITEMS _ '(("Add item" :ADD "Add a new entry to the who-line")
                                          ("Remove item" :REMOVE "Remove an existing entry from the who-line"))
                                TITLE _ "Change WHO-LINE entries"))
                  (:ADD (LET* [[ITEMS (for entry in *WHO-LINE-ENTRY-REGISTRY* unless (MEMBER entry *WHO-LINE-ENTRIES*
                                                                                       )
                                             collect `(,(ENTRY-DESCRIPTION entry)
                                                       ',entry]
                               (NEW-ENTRY (if ITEMS
                                              then (MENU (create MENU
                                                                 ITEMS _ ITEMS
                                                                 TITLE _ "Entry to add to WHO-LINE"]
                              (if NEW-ENTRY
                                  then (SETQ *WHO-LINE-ENTRIES* (CONS NEW-ENTRY *WHO-LINE-ENTRIES*))
                                       (INSTALL-WHO-LINE-OPTIONS))))
                  (:REMOVE (LET* [[ITEMS (for entry in *WHO-LINE-ENTRIES* collect `(,(ENTRY-DESCRIPTION entry)
                                                                                   ',entry]
                                  (BAD-ENTRY (if ITEMS
                                                 then (MENU (create MENU
                                                                    ITEMS _ ITEMS
                                                                    TITLE _ "Entry to remove from WHO-LINE"]
                                 (if BAD-ENTRY
                                     then (SETQ *WHO-LINE-ENTRIES* (CL:REMOVE BAD-ENTRY *WHO-LINE-ENTRIES*))
                                          (INSTALL-WHO-LINE-OPTIONS))))))])
```

(WHO-LINE-COPY-INSERT

```
  [LAMBDA (X)                                                    ; Edited 18-Mar-87 13:11 by smL
    (LET [(TTY-WINDOW (WFROMDS (PROCESS.TTY (TTY.PROCESS NIL]
         (if [OR (IMAGEOBJP X)
                 (AND (WINDOWP TTY-WINDOW)
                      (WINDOWPROP TTY-WINDOW 'COPYINSERTFN]
             then (COPYINSERT X)
           else (BKSYSBUF X NIL])

)

(DEFINEQ
```

(WHO-LINE-REDISPLAY-INTERRUPT

```
  [LAMBDA NIL                                                    ; Edited 20-Apr-87 11:32 by smL
```

;;; Update the current who-line window because the user has requested it via an interrupt.

```
    (if (BOUNDP '*WHO-LINE*)
        then ;; Update the Who-Line, if it is available
             (WITH-AVAILABLE-WHO-LINE *WHO-LINE* (if (AND (OPENWP *WHO-LINE*)
                                                          (WINDOWPROP *WHO-LINE* 'VALID))
                                                     then ; Flash the Who-line to let people know that it is being updated
                                                          (CLOSEW *WHO-LINE*)
                                                          (OPENW *WHO-LINE*)
                                                          ; The update the entries
                                                          (UPDATE-WHO-LINE *WHO-LINE* (WINDOWPROP *WHO-LINE*
                                                                                                  'ENTRIES])

)


(DEFGLOBALVAR *WHO-LINE* NIL
   "The who-line window")


(DEFGLOBALVAR *WHO-LINE-UPDATE-TIMER* NIL
   "Timer for controlling updating of the Who-Line")

(DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE

(APPENDTOVAR AFTERSYSOUTFORMS (INSTALL-WHO-LINE-OPTIONS))

(APPENDTOVAR AFTERMAKESYSFORMS (INSTALL-WHO-LINE-OPTIONS))

(APPENDTOVAR \SYSTEMTIMERVARS (*WHO-LINE-UPDATE-TIMER* TICKS))
)


(DEFMACRO INVERT-WHO-LINE-ENTRY (ENTRY WINDOW)
   `(WITH WHO-LINE-ENTRY ,ENTRY (BLTSHADE BLACKSHADE ,WINDOW NAME-START 0 (DIFFERENCE VALUE-END NAME-START)
                                          NIL
                                          'INVERT)
          (CHANGE INVERTED? (NOT INVERTED?))))

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE
```

```
(RECORD WHO-LINE-ENTRY (NAME FORM NAME-START VALUE-START VALUE-END PREV-VALUE WHEN-SELECTED-FN INVERTED?
                              RESET-FORM DESCRIPTION))
)
)
```

;; Macros that lets us lock down the Who-Line while we evaluate some forms

```
(DEFMACRO WITH-WHO-LINE (WHO-LINE &BODY FORMS)
   "Evaluate the forms with the who-line locked down"
   `(WITH.MONITOR (WINDOWPROP ,WHO-LINE 'LOCK)
           ,@FORMS))


(DEFMACRO WITH-AVAILABLE-WHO-LINE (WHO-LINE &BODY FORMS)
   "Evaluate the forms with the who-line locked down, if the who-line is available"
   [LET ((LOCK (CL:GENSYM)))
        `(LET ((,LOCK (OBTAIN.MONITORLOCK (WINDOWPROP ,WHO-LINE 'LOCK)
                              T)))
              (CL:UNWIND-PROTECT                                    ; Only eval the forms if we got the lock
                  (COND
                     (,LOCK ,@FORMS))
                  ;; Now for the cleanup forms
                  [COND
                     ((EQ ,LOCK T)                                  ; Had the lock before, so no need to release it
                      NIL)
                     ((NULL ,LOCK)                                  ; Couldn't get the lock, so no need to release it
                      NIL)
                     (T                                             ; We got the lock, and need to release it
                        (RELEASE.MONITORLOCK ,LOCK])])


;;; -------------------------------------------------------------------

;;; Initialize the who-line

(DECLARE%: DONTEVAL@LOAD DONTEVAL@COMPILE

(INSTALL-WHO-LINE-OPTIONS)

(ADDTOVAR BACKGROUNDFNS PERIODICALLY-UPDATE-WHO-LINE)
)


;;; -------------------------------------------------------------------

;;; Filemanager stuff

(DECLARE%: DONTCOPY

(PUTPROPS WHO-LINE MAKEFILE-ENVIRONMENT (:PACKAGE "IL" :READTABLE "INTERLISP" :BASE 10))

(PUTPROPS WHO-LINE FILETYPE :COMPILE-FILE)
)
```

FUNCTION INDEX

CHANGE-TTY-PROCESS-INTERACTIVELY 8
CURRENT-PROFILE 14
CURRENT-TTY-PACKAGE 7
CURRENT-TTY-READTABLE-NAME 8
INSTALL-WHO-LINE-OPTIONS 3
PERIODICALLY-UPDATE-WHO-LINE 17
REDISPLAY-WHO-LINE 16
SET-CONNECTED-DIRECTORY-INTERACTIVELY 9
SET-PACKAGE-INTERACTIVELY 7
SET-PROFILE-INTERACTIVELY 14
SET-READTABLE-INTERACTIVELY 8
SET-TTY-PACKAGE-INTERACTIVELY 7
SET-TTY-PROFILE-INTERACTIVELY 14
SET-TTY-READTABLE-INTERACTIVELY 8
SETUP-WHOLINE-TIMER 17
UPDATE-WHO-LINE 17
WHEN-WHO-LINE-SELECTED-FN 18
WHO-LINE-CHANGE-USER 5
WHO-LINE-CONTROL-SELECT 18
WHO-LINE-COPY-INSERT 19
WHO-LINE-CURRENT-DIRECTORY 9
WHO-LINE-HOST-NAME 6
WHO-LINE-REDISPLAY-INTERRUPT 19
WHO-LINE-SAVE-VMEM 10
WHO-LINE-SET-TIME 12
WHO-LINE-SHOW-ACTIVE 12
WHO-LINE-SYMBOL-SPACE 11
WHO-LINE-TIME 11
WHO-LINE-TTY-PROCESS 8
WHO-LINE-TTY-STATE 14
WHO-LINE-USER-AFTER-LOGIN 6
WHO-LINE-USERNAME 5
WHO-LINE-VMEM 10
WHO-LINE-WHAT-IS-RUNNING 15
\PERIODICALLY-WHO-LINE-SHOW-ACTIVE 13
\UPDATE-WHO-LINE-ACTIVE-FLAG 13

VARIABLE INDEX

WHO-LINE 19
WHO-LINE-ACTIVE-PERIOD 13
WHO-LINE-ACTIVE-TIMER 13
WHO-LINE-ANCHOR 16
WHO-LINE-BORDER 16
WHO-LINE-COLOR 16
WHO-LINE-CURRENT-USER 6
WHO-LINE-DIRECTORIES 10
WHO-LINE-DIRECTORY-ENTRY 10
WHO-LINE-DISPLAY-NAMES? 16
WHO-LINE-ENTRIES 15
WHO-LINE-ENTRY-REGISTRY 16
WHO-LINE-HOST-NAME 6
WHO-LINE-HOST-NAME-ENTRY 6
WHO-LINE-LAST-DIRECTORY 10
WHO-LINE-LAST-VMEM 11
WHO-LINE-NAME-FONT 16
WHO-LINE-OLD-TIME 12
WHO-LINE-PACKAGE-ENTRY 7
WHO-LINE-PACKAGE-NAME-CACHE 7
WHO-LINE-PROFILE-ENTRY 14
WHO-LINE-READTABLE-ENTRY 8
WHO-LINE-SHOW-ACTIVE-ENTRY 13
WHO-LINE-STATE 15
WHO-LINE-STATE-UNINTERESTING-FNS 15
WHO-LINE-SYMBOL-SPACE 11
WHO-LINE-SYMBOL-SPACE-ENTRY 11
WHO-LINE-TIME-ENTRY 12
WHO-LINE-TIMER 12
WHO-LINE-TITLE 16
WHO-LINE-TTY-PROC-ENTRY 9
WHO-LINE-TTY-STATE-ENTRY 15
WHO-LINE-UPDATE-INTERVAL 16
WHO-LINE-UPDATE-TIMER 19
WHO-LINE-USER-ENTRY 6
WHO-LINE-VALUE-FONT 16
WHO-LINE-VMEM-ENTRY 11
BACKGROUNDFNS 20
\AFTERLOGINFNS 6
\SYSTEMCACHEVARS 6

PROPERTY INDEX

AWAIT.EVENT 15
BLOCK 15
EXCHANGEPUPS 15
GETPUP 15
PUTSEQUIN 15
SENDPUP 15
WAIT.FOR.TTY 15
WHO-LINE 20
\LEAF.READPAGES ..15
\SENDLEAF 15
\TTYBACKGROUND ...15
\WAITFORSYSBUFP ..15
\\getkey 15

MACRO INDEX

INVERT-WHO-LINE-ENTRY 19
WITH-AVAILABLE-WHO-LINE 20
WITH-WHO-LINE 20

RECORD INDEX

WHO-LINE-ENTRY 20