

File created: 9-Apr-2024 12:59:40 {DSK}<home>larry>il>medley>sources>CMLPATHNAME.;2

edit by: lmm

previous date: 23-Mar-2024 22:31:11 {DSK}<home>larry>il>medley>sources>CMLPATHNAME.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

(RPAQQ **CMLPATHNAMECOMS**

[;; Common Lisp pathname functions

(PROP FILETYPE CMLPATHNAME)

(COMS ;; useful macros

(FUNCTIONS %%WILD-NAME %%COMPONENT-STRING))

(STRUCTURES PATHNAME DIRECTORY-COMPONENT)

(FNS %%PRINT-PATHNAME CL:MAKE-PATHNAME %%PRINT-DIRECTORY-COMPONENT)

(FUNCTIONS CL:PATHNAME-HOST CL:PATHNAME-DEVICE CL:PATHNAME-DIRECTORY CL:PATHNAME-NAME CL:PATHNAME-TYPE
CL:PATHNAME-VERSION)

(FNS PATHNAME CL:MERGE-PATHNAMES FILE-NAME CL:HOST-NAMESTRING CL:ENOUGH-NAMESTRING %%NUMERIC-STRING-P)

(FUNCTIONS CL:NAMESTRING CL:PARSE-NAMESTRING CL:TRUENAME)

(FUNCTIONS %%MAKE-PATHNAME)

(FUNCTIONS %%PATHNAME-EQUAL %%DIRECTORY-COMPONENT-EQUAL)

(FUNCTIONS %%INITIALIZE-DEFAULT-PATHNAME)

(VARIABLES *DEFAULT-PATHNAME-DEFAULTS*)

(COMS ;; Interlisp-D compatibility

(FUNCTIONS INTERLISP-NAMESTRING UNPACKPATHNAME.STRING))

(FUNCTIONS CL:FILE-NAMESTRING CL:DIRECTORY-NAMESTRING)

(DECLARE%: DONTVAL@LOAD DOCOPY (P (%%INITIALIZE-DEFAULT-PATHNAME)))

(DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILEVAR (ADDVARS (NLAMA)

(NLAML)

(LAMA CL:ENOUGH-NAMESTRING

CL:MERGE-PATHNAMES

CL:MAKE-PATHNAME

%%PRINT-PATHNAME])

;; Common Lisp pathname functions

(PUTPROPS **CMLPATHNAME FILETYPE** CL:COMPILE-FILE)

;; useful macros

(DEFMACRO **%%WILD-NAME** (STRING)

`(LET ((S ,STRING))

(CL:IF (STRING-EQUAL S "**")

:WILD

S)))

(DEFMACRO **%%COMPONENT-STRING** (COMPONENT)

`(MKSTRING (OR ,COMPONENT "")))

(CL:DEFSTRUCT (**PATHNAME** (:CONC-NAME %%PATHNAME-)

(:PRINT-FUNCTION CL:PRINT-PATHNAME)

(:CONSTRUCTOR %%%MAKE-PATHNAME)

(:PREDICATE CL:PATHNAMEP))

HOST

DEVICE

DIRECTORY

NAME

TYPE

VERSION)

(CL:DEFSTRUCT (**DIRECTORY-COMPONENT** (:CONC-NAME %%DIRECTORY-COMPONENT-)

(:PRINT-FUNCTION %%PRINT-DIRECTORY-COMPONENT)

(:CONSTRUCTOR %%MAKE-DIRECTORY-COMPONENT)

(:PREDICATE %%DIRECTORY-COMPONENT-P))

TYPE

PATH)

(DEFINEQ

(**%%PRINT-PATHNAME**

(CL:LAMBDA (S STREAM D)

(**DECLARE** (IGNORE D))

(CL:FORMAT STREAM "#P~S" (CL:NAMESTRING S))))

; Edited 23-Mar-2024 22:25 by lmm
(* hdj "19-Sep-86 15:49")

(CL:MAKE-PATHNAME

```
(CL:LAMBDA (&KEY DEFAULTS (HOST NIL HOSTP)
              (DEVICE NIL DEVICEP)
              (DIRECTORY NIL DIRECTORYP)
              (NAME NIL NAMEP)
              (TYPE NIL TYPEP)
              (VERSION NIL VERSIONP))
```

; Edited 28-Sep-90 15:02 by jds

:: Create a pathname from host, device, directory, name, type and version. If any field is omitted, it is obtained from defaults as though by
 :: merge-pathnames.

```
(CL:IF DEFAULTS
  [LET ((DEFAULTS (PATHNAME DEFAULTS)))
    (CL:UNLESS HOSTP
      (SETQ HOST (%PATHNAME-HOST DEFAULTS)))
    (CL:UNLESS DEVICEP
      (SETQ DEVICE (%PATHNAME-DEVICE DEFAULTS)))
    (CL:UNLESS DIRECTORYP
      (SETQ DIRECTORY (%PATHNAME-DIRECTORY DEFAULTS)))
    (CL:UNLESS NAMEP
      (SETQ NAME (%PATHNAME-NAME DEFAULTS)))
    (CL:UNLESS TYPEP
      (SETQ TYPE (%PATHNAME-TYPE DEFAULTS)))
    (CL:UNLESS VERSIONP
      (SETQ VERSION (%PATHNAME-VERSION DEFAULTS))))]
  (CL:UNLESS HOSTP
    (SETQ HOST (%PATHNAME-HOST *DEFAULT-PATHNAME-DEFAULTS*)))
  (%%MAKE-PATHNAME (CL:IF (STRINGP HOST)
                          (COERCE HOST 'CL:SIMPLE-STRING)
                          HOST)
    (CL:IF (STRINGP DEVICE)
      (COERCE DEVICE 'CL:SIMPLE-STRING)
      DEVICE)
    (CL:IF DIRECTORY
      (CL:TYPECASE DIRECTORY
        (DIRECTORY-COMPONENT DIRECTORY)
        ((OR CL:SYMBOL STRING) [COND
          ((AND (CL:SYMBOLP DIRECTORY)
                (EQ DIRECTORY :WILD))
            (%%MAKE-DIRECTORY-COMPONENT :TYPE :DIRECTORY :PATH :WILD))
          (T (SETQ DIRECTORY (STRING DIRECTORY))
              (LET [(DEFAULT-DIR (CL:IF DEFAULTS
                                         (%PATHNAME-DIRECTORY DEFAULTS)
                                         (%PATHNAME-DIRECTORY
                                          *DEFAULT-PATHNAME-DEFAULTS*))
                    )
                (DIREND (CL:1- (CL:LENGTH DIRECTORY))
                  (CASE (CL:CHAR DIRECTORY DIREND)
                    ((#\> #\/)
                     ; MAKE-PATHNAME does not accept :SUBDIRECTORY
                     ; argument. Thus a subdirectory and a relative directory is
                     ; indicated with the trail directory delimiter.
                     ;; If HOST is also specified, it is a relative directory, otherwise a
                     ;; subdirectory.
                     (CL:IF HOSTP
                       (%%MAKE-DIRECTORY-COMPONENT :TYPE :RELATIVE
                                                     :PATH (CL:SUBSEQ DIRECTORY 0 DIREND)
                                                     )
                       (%%MAKE-DIRECTORY-COMPONENT
                        :TYPE :DIRECTORY :PATH
                        (CL:CONCATENATE 'STRING (
                                         %DIRECTOR-COMPONENT-PATH
                                         DEFAULT-DIR)
                                         (CL:SECOND \FILENAME.SYNTAX)
                                         (CL:SUBSEQ DIRECTORY 0 DIREND))))))
                    (T (%%MAKE-DIRECTORY-COMPONENT :TYPE :DIRECTORY
                                                     :PATH DIRECTORY))))))
          (T DIRECTORY))
        DIRECTORY)
      (CL:IF (STRINGP NAME)
        (COERCE NAME 'CL:SIMPLE-STRING)
        NAME)
      (CL:IF (STRINGP TYPE)
        (COERCE TYPE 'CL:SIMPLE-STRING)
        TYPE)
      VERSION)))
```

(%%PRINT-DIRECTORY-COMPONENT

```
(CL:LAMBDA (S STREAM D)
  (DECLARE (IGNORE D))
```

; Edited 7-Mar-90 17:59 by nm

```
#|(CL:FORMAT STREAM "#.(~S~S)" (QUOTE DIRECTORY-COMPONENT) (CASE (%%DIRECTORY-COMPONENT-TYPE S) ((:SUBDIRECTORY
:RELATIVE) (CL:CONCATENATE (QUOTE STRING) (%%DIRECTORY-COMPONENT-PATH S) ">")) (T (CL:CONCATENATE (QUOTE STRING)
(CL:FIRST \FILENAME.SYNTAX) (%%DIRECTORY-COMPONENT-PATH S) (CL:SECOND \FILENAME.SYNTAX)))))|#
```

```
(LET ((PATH (%%DIRECTORY-COMPONENT-PATH S)))
```

```

(CL:FORMAT STREAM "~A" (CASE (%DIRECTORY-COMPONENT-TYPE S)
  ((:SUBDIRECTORY :RELATIVE) (CL:CONCATENATE 'STRING PATH ">"))
  (T (CL:IF (EQ PATH :WILD)
    (CL:CONCATENATE 'STRING (CL:FIRST \FILENAME.SYNTAX)
      "*"
      (CL:SECOND \FILENAME.SYNTAX))
    (CL:CONCATENATE 'STRING (CL:FIRST \FILENAME.SYNTAX)
      PATH
      (CL:SECOND \FILENAME.SYNTAX))))))
)

```

```

(CL:DEFUN CL:PATHNAME-HOST (PATHNAME)
  ;; takes a stream, string, symbol, or pathname as arg, and returns the host slot of it
  (%PATHNAME-HOST (PATHNAME PATHNAME)))

```

```

(CL:DEFUN CL:PATHNAME-DEVICE (PATHNAME)
  ;; takes a stream, string, symbol, or pathname as arg, and returns the device slot of it
  (%PATHNAME-DEVICE (PATHNAME PATHNAME)))

```

```

(CL:DEFUN CL:PATHNAME-DIRECTORY (PATHNAME)
  ;; takes a stream, string, symbol, or pathname as arg, and returns the directory slot of it
  (%PATHNAME-DIRECTORY (PATHNAME PATHNAME)))

```

```

(CL:DEFUN CL:PATHNAME-NAME (PATHNAME)
  ;; takes a stream, string, symbol, or pathname as arg, and returns the name slot of it
  (%PATHNAME-NAME (PATHNAME PATHNAME)))

```

```

(CL:DEFUN CL:PATHNAME-TYPE (PATHNAME)
  ;; takes a stream, string, symbol, or pathname as arg, and returns the type slot of it
  (%PATHNAME-TYPE (PATHNAME PATHNAME)))

```

```

(CL:DEFUN CL:PATHNAME-VERSION (PATHNAME)
  ;; takes a stream, string, symbol, or pathname as arg, and returns the version slot of it
  (%PATHNAME-VERSION (PATHNAME PATHNAME)))

```

```

(DEFINEQ

```

```

(PATHNAME

```

```

  (CL:LAMBDA (THING)

```

```

    ; Edited 1-May-2023 07:04 by Imm
    (* hdj " 2-Apr-86 11:01")

```

```

    ;; Turns Thing into a pathname. Thing may be a string, symbol, stream, or pathname.

```

```

    [CL:CHECK-TYPE THING (OR STRING STREAM PATHNAME (AND CL:SYMBOL (NOT NULL))
    (CL:VALUES (CL:PARSE-NAMESTRING THING)))]

```

```

(CL:MERGE-PATHNAMES

```

```

  [CL:LAMBDA (PATHNAME &OPTIONAL (DEFAULTS *DEFAULT-PATHNAME-DEFAULTS*)
    (DEFAULT-VERSION :NEWEST CL::VERSION-SPECIFIED-P))

```

```

    ; Edited 21-Aug-90 17:12 by nm

```

```

;;; Merge-Pathnames -- Public Returns a new pathname whose fields are the same as the fields in PATHNAME except that NIL fields are filled in from
;;; defaults. Type and Version field are only done if name field has to be done (see manual for explanation). Fills in unspecified slots of Pathname from
;;; Defaults (defaults to *default-pathname-defaults*). If the version remains unspecified, gets it from Default-Version.

```

```

(LET* ((PATH (PATHNAME PATHNAME))
  (DEFAULT-PATH (PATHNAME DEFAULTS))
  (HOST (OR (%PATHNAME-HOST PATH)
    (%PATHNAME-HOST DEFAULT-PATH)))
  (NAME (%PATHNAME-NAME PATH))
  (DEVICE (%PATHNAME-DEVICE PATH))
  (DIR (%PATHNAME-DIRECTORY PATH))
  (DEFAULT-DIR (%PATHNAME-DIRECTORY DEFAULT-PATH))
  DIREND DEFAULT-TYPE)
  (%MAKE-PATHNAME HOST (OR DEVICE (%PATHNAME-DEVICE DEFAULT-PATH))
    (OR [AND DIR DEFAULT-DIR (CASE (%DIRECTORY-COMPONENT-TYPE DIR)
      (:SUBDIRECTORY
        (CASE (SETQ DEFAULT-TYPE (%DIRECTORY-COMPONENT-TYPE
          DEFAULT-DIR))
          (:SUBDIRECTORY
            ; Default is also a subdirectory, so explicit subdire overrides it
            DIR)
          (T ; Default is a full directory or a relative directory. Make sure to
            ; keep the type of the directory being same as the default one.

```

```

(CL:IF (EQ (%%DIRECTORY-COMPONENT-PATH DEFAULT-DIR)
:WILD)
(%MAKE-DIRECTORY-COMPONENT :TYPE :RELATIVE
:PATH (%%DIRECTORY-COMPONENT-PATH DIR))
(%MAKE-DIRECTORY-COMPONENT :TYPE DEFAULT-TYPE
:PATH (CL:CONCATENATE 'STRING
(%%DIRECTORY-COMPONENT-PATH
DEFAULT-DIR)
(CL:SECOND \FILENAME.SYNTAX
)
(%%DIRECTORY-COMPONENT-PATH
DIR))))))
(T (CL:IF (NOT (EQ (%%DIRECTORY-COMPONENT-PATH DIR)
:WILD))
DIR
DEFAULT-DIR))))
DIR DEFAULT-DIR)
(OR NAME (%%PATHNAME-NAME DEFAULT-PATH))
(OR (%%PATHNAME-TYPE PATH)
(%%PATHNAME-TYPE DEFAULT-PATH))
(OR (%%PATHNAME-VERSION PATH)
(CL:IF NAME
(CL:IF CL::VERSION-SPECIFIED-P
DEFAULT-VERSION
:NEWEST)
(OR (%%PATHNAME-VERSION DEFAULT-PATH)
(CL:IF CL::VERSION-SPECIFIED-P
DEFAULT-VERSION
:NEWEST))))))

```

(FILE-NAME

```

[CL:LAMBDA (FILE) (* hdj "9-Oct-86 15:12")
(LET ((NAME (FULLNAME FILE)))
(if (STREAMP NAME)
then ""
else (MKSTRING NAME]))

```

(CL:HOST-NAMESTRING

```

[CL:LAMBDA (PATHNAME) (* hdj "11-Jun-86 11:29")
;; Returns the host part of PATHNAME as a string.
(%%COMPONENT-STRING (%%PATHNAME-HOST (PATHNAME PATHNAME)))

```

(CL:ENOUGH-NAMESTRING

```

[CL:LAMBDA (PATHNAME &OPTIONAL (DEFAULTS *DEFAULT-PATHNAME-DEFAULTS*))
; Edited 7-Mar-90 16:49 by nm
;; Enough-Namestring returns a string which uniquely identifies PATHNAME w.r.t. DEFAULTS.

```

```

(LET* ((*PRINT-BASE* 10)
(PATH (PATHNAME PATHNAME))
(DEFAULT-PATHNAME (PATHNAME DEFAULTS))
(HOST (%%PATHNAME-HOST PATH))
(DEVICE (%%PATHNAME-DEVICE PATH))
(DIRECTORY (%%PATHNAME-DIRECTORY PATH))
(NAME (%%PATHNAME-NAME PATH))
(TYPE (%%PATHNAME-TYPE PATH))
(VERSION (%%PATHNAME-VERSION PATH))
(RESULT ""))
(NEED-NAME NIL))
(CL:WHEN [AND HOST (CL:STRING-NOT-EQUAL HOST (%%COMPONENT-STRING (%%PATHNAME-HOST
DEFAULT-PATHNAME)
""))])
(CL:WHEN [AND DEVICE (CL:STRING-NOT-EQUAL DEVICE (%%COMPONENT-STRING (%%PATHNAME-DEVICE
DEFAULT-PATHNAME)
""))])
(CL:WHEN [AND DIRECTORY (NOT (%%DIRECTORY-COMPONENT-EQUAL DIRECTORY (%%PATHNAME-DIRECTORY
DEFAULT-PATHNAME)
[CL:SETQ RESULT (CASE (%%DIRECTORY-COMPONENT-TYPE DIRECTORY)
((SUBDIRECTORY :RELATIVE)
; The initial directory delimiter is not needed for a subdirectory
; and a relative directory. Just concatenate a trail directory
; delimiter.
(CL:CONCATENATE 'CL:SIMPLE-STRING RESULT (%%DIRECTORY-COMPONENT-PATH
DIRECTORY)
(CL:SECOND \FILENAME.SYNTAX))
(T (CL:IF (EQ (%%DIRECTORY-COMPONENT-PATH DIRECTORY)
:WILD)
(CL:CONCATENATE 'CL:SIMPLE-STRING RESULT (CL:FIRST
\FILENAME.SYNTAX)
" * "
(CL:SECOND \FILENAME.SYNTAX))
(CL:CONCATENATE 'CL:SIMPLE-STRING RESULT (CL:FIRST

```

```

                                \FILENAME.SYNTAX)
                                (%%DIRECTORY-COMPONENT-PATH DIRECTORY)
                                (CL:SECOND \FILENAME.SYNTAX))))))
(CL:WHEN [AND NAME (CL:STRING-NOT-EQUAL NAME (%%COMPONENT-STRING (%%PATHNAME-NAME
                                DEFAULT-PATHNAME]
                                (CL:SETQ RESULT (CL:CONCATENATE 'CL:SIMPLE-STRING RESULT (CL:PRINC-TO-STRING NAME))))
(CL:WHEN [AND TYPE (CL:STRING-NOT-EQUAL TYPE (%%COMPONENT-STRING (%%PATHNAME-TYPE
                                DEFAULT-PATHNAME]
                                (SETQ RESULT (CL:CONCATENATE 'CL:SIMPLE-STRING RESULT "." (CL:PRINC-TO-STRING TYPE))))
(CL:WHEN [AND VERSION (OR NEED-NAME (CL:STRING-NOT-EQUAL (CL:PRINC-TO-STRING VERSION)
                                (%%PATHNAME-VERSION DEFAULT-PATHNAME]
                                [SETQ RESULT (CL:CONCATENATE 'CL:SIMPLE-STRING RESULT (CASE VERSION
                                (:WILD ";"*)
                                (:(NEWEST NIL) "")
                                (CL:OTHERWISE (CL:CONCATENATE
                                'CL:SIMPLE-STRING
                                ";" (
                                CL:PRINC-TO-STRING
                                VERSION)))]))
                                RESULT)))

```

(%%NUMERIC-STRING-P

(* hdj "28-Jul-86 12:25")

```

[LAMBDA (STRING)
  (AND (CL:STRINGP STRING)
    (for CHAR instring STRING do (if (OR (ILESSP CHAR (CHARCODE 0))
                                          (IGREATERP CHAR (CHARCODE 9)))
      then (RETURN NIL))
    finally (RETURN T)))
)

```

(CL:DEFUN CL:NAMESTRING (PATHNAME)

;;; Returns the full form of PATHNAME as a string.

```

(CL:WHEN (AND (STREAMP PATHNAME)
              (NOT (fetch (STREAM NAMEDP) of PATHNAME))) ; unnamed streams have the empty string as name.
(CL:RETURN-FROM CL:NAMESTRING ""))
[LET* ((PATHNAME (PATHNAME PATHNAME))
      (CL::HOST (%%PATHNAME-HOST PATHNAME))
      (CL::DEVICE (%%PATHNAME-DEVICE PATHNAME))
      (CL::DIRECTORY (%%PATHNAME-DIRECTORY PATHNAME))
      (CL::NAME (%%PATHNAME-NAME PATHNAME))
      (TYPE (%%PATHNAME-TYPE PATHNAME))
      (CL::VERSION (%%PATHNAME-VERSION PATHNAME))
      (CL::RESULT NIL))
(CONCATLIST (NCONC (CL:WHEN CL::HOST (LIST "{" CL::HOST "}"))
                  (CL:WHEN CL::DEVICE (LIST CL::DEVICE ":"))
                  (CL:WHEN CL::DIRECTORY
                    (CASE (%%DIRECTORY-COMPONENT-TYPE CL:DIRECTORY)
                      (:(SUBDIRECTORY :RELATIVE) ; The initial directory delimiter is not needed for a subdirectory
                                                    ; and a releative directory. Just concatenate a trail directory
                                                    ; delimiter.
                      (LIST (%%DIRECTORY-COMPONENT-PATH CL:DIRECTORY)
                           (CL:SECOND \FILENAME.SYNTAX)))
                    (T (CL:IF (EQ (%%DIRECTORY-COMPONENT-PATH CL:DIRECTORY)
                                :WILD)
                              NIL
                              (LIST (CL:FIRST \FILENAME.SYNTAX)
                                   (%%DIRECTORY-COMPONENT-PATH CL:DIRECTORY)
                                   (CL:SECOND \FILENAME.SYNTAX))))))
                  (CL:WHEN CL::NAME
                    (LIST (CL:IF (EQ CL::NAME :WILD)
                                "*)"
                           CL::NAME)))
                  (CL:WHEN TYPE
                    (LIST "." (CL:IF (EQ TYPE :WILD)
                                "*)"
                                TYPE)))
                  (CL:WHEN (AND CL::VERSION (OR (NOT (EQ CL::VERSION ':(NEWEST))
                                                  CL::NAME TYPE))
                    [COND
                      [[AND (EQ \MACHINETYPE \MAIKO)
                          (STREQUAL "UNIX" (U-CASE (MKSTRING CL::HOST)
                                                    ; {UNIX} device on Maiko breaks the Interlisp-D original file
                                                    ; naming convention. The trail semicolon is regarded as a part
                                                    ; of the file name rather than a "highest versioned" file! Thus, if
                                                    ; :newest, we have to elimit the semicolon.
                          (CASE CL::VERSION
                            (:(WILD)
                              (LIST (CL:THIRD \FILENAME.SYNTAX)
                                   "*))
                            (:(NEWEST) (LIST ""))
                            (T (LIST (CL:THIRD \FILENAME.SYNTAX)
                                   CL::VERSION)))]

```

```
(CL:DEFUN %%PATHNAME-EQUAL (PATHNAME1 PATHNAME2)
  (AND (CL:EQUAL (%%PATHNAME-HOST PATHNAME1)
```

```

    (%%PATHNAME-HOST PATHNAME2))
  (CL:EQUAL (%%PATHNAME-DEVICE PATHNAME1)
    (%%PATHNAME-DEVICE PATHNAME2))
  (%%DIRECTORY-COMPONENT-EQUAL (%%PATHNAME-DIRECTORY PATHNAME1)
    (%%PATHNAME-DIRECTORY PATHNAME2))
  (CL:EQUAL (%%PATHNAME-NAME PATHNAME1)
    (%%PATHNAME-NAME PATHNAME2))
  (CL:EQUAL (%%PATHNAME-TYPE PATHNAME1)
    (%%PATHNAME-TYPE PATHNAME2))
  (CL:EQUAL (%%PATHNAME-VERSION PATHNAME1)
    (%%PATHNAME-VERSION PATHNAME2)))

(CL:DEFUN %%DIRECTORY-COMPONENT-EQUAL (COMPONENT1 COMPONENT2)
  (CL:IF (AND (%%DIRECTORY-COMPONENT-P COMPONENT1)
    (%%DIRECTORY-COMPONENT-P COMPONENT2))
    (AND (CL:EQUAL (%%DIRECTORY-COMPONENT-TYPE COMPONENT1)
      (%%DIRECTORY-COMPONENT-TYPE COMPONENT2))
      (CL:EQUAL (%%DIRECTORY-COMPONENT-PATH COMPONENT1)
        (%%DIRECTORY-COMPONENT-PATH COMPONENT2)))
    (CL:EQUAL COMPONENT1 COMPONENT2)))

(CL:DEFUN %%INITIALIZE-DEFAULT-PATHNAME ()
  (DECLARE (GLOBALVARS *DEFAULT-PATHNAME-DEFAULTS* \CONNECTED.DIRECTORY))
  (if (NOT (BOUNDP '\CONNECTED.DIRECTORY))
    then (SETQ \CONNECTED.DIRECTORY '{DSK})))
  [SETQ *DEFAULT-PATHNAME-DEFAULTS* (CL:PARSE-NAMESTRING \CONNECTED.DIRECTORY (FILENAMEFIELD
    \CONNECTED.DIRECTORY
    'HOST]

  (CL:SETF (%%PATHNAME-VERSION *DEFAULT-PATHNAME-DEFAULTS*)
    :NEWEST)
  *DEFAULT-PATHNAME-DEFAULTS*)

(CL:DEFVAR *DEFAULT-PATHNAME-DEFAULTS*)

;; Interlisp-D compatibility

(CL:DEFUN INTERLISP-NAMESTRING (PATHNAME)
  ;; Returns the full form of PATHNAME as an Interlisp string.

  (MKSTRING (CL:NAMESTRING PATHNAME)))

(CL:DEFUN UNPACKPATHNAME.STRING (FILE &OPTIONAL ONEFIELDFLG DIRFLG ATOMFLG)
  ;; Simulate the action of UNPACKFILENAME.STRING on a pathname
  ;;
  (DECLARE (IGNORE DIRFLG))
  [if ONEFIELDFLG
    then [AND (CL:CONSP ONEFIELDFLG)
      (SETQ ONEFIELDFLG (CAR (CL:INTERSECTION ONEFIELDFLG '(HOST DEVICE DIRECTORY NAME EXTENSION
        VERSION]

      (LET [(RESULT (CASE ONEFIELDFLG
        (HOST (CL:PATHNAME-HOST FILE))
        (DEVICE (CL:PATHNAME-DEVICE FILE))
        (DIRECTORY (CL:PATHNAME-DIRECTORY FILE))
        (NAME (CL:PATHNAME-NAME FILE))
        (EXTENSION (CL:PATHNAME-TYPE FILE))
        (VERSION (CL:PATHNAME-VERSION FILE))
        (CL:OTHERWISE NIL))]
        (if ATOMFLG
          then (MKATOM RESULT)
          else RESULT))
    else (LET ((COMPONENT))
      (APPEND (if (SETQ COMPONENT (CL:PATHNAME-HOST FILE))
        then (LIST 'HOST (if ATOMFLG
          then (MKATOM COMPONENT)
          else COMPONENT)
        COMPONENT))
      (if (SETQ COMPONENT (CL:PATHNAME-DEVICE FILE))
        then (LIST 'DEVICE (if ATOMFLG
          then (MKATOM COMPONENT)
          else COMPONENT)))
      (if (SETQ COMPONENT (CL:PATHNAME-DIRECTORY FILE))
        then (LIST 'DIRECTORY (if ATOMFLG
          then (MKATOM COMPONENT)
          else COMPONENT)))
      (if (SETQ COMPONENT (CL:PATHNAME-NAME FILE))
        then (LIST 'NAME (if ATOMFLG
          then (MKATOM COMPONENT)
          else COMPONENT)))
      (if (SETQ COMPONENT (CL:PATHNAME-TYPE FILE))

```

```

    then (LIST 'EXTENSION (if ATOMFLG
                                then (MKATOM COMPONENT)
                                else COMPONENT)))
    (if (SETQ COMPONENT (CL:PATHNAME-VERSION FILE))
        then (LIST 'VERSION (if ATOMFLG
                                then (MKATOM COMPONENT)
                                else (MKSTRING COMPONENT)))

(CL:DEFUN CL:FILE-NAMESTRING (PATHNAME)
  (LET* ((*PRINT-BASE* 10)
         (*PRINT-RADIX* NIL)
         (PATH (PATHNAME PATHNAME))
         [RESULT (CL:CONCATENATE 'CL:SIMPLE-STRING (MKSTRING (%%COMPONENT-STRING (%%PATHNAME-NAME PATH)))
                                " "
                                (MKSTRING (%%COMPONENT-STRING (%%PATHNAME-TYPE PATH)
                                (VERSION (%%PATHNAME-VERSION PATH)))
         (CL:WHEN VERSION
           [SETQ RESULT (CL:CONCATENATE 'CL:SIMPLE-STRING RESULT (CASE VERSION
                                                                           (:WILD ";"*)
                                                                           (:NEWEST ";"")
                                                                           (CL:OTHERWISE (CL:CONCATENATE
                                                                           'CL:SIMPLE-STRING ";"
                                                                           (CL:PRINC-TO-STRING
                                                                           VERSION))))])])
        RESULT))

(CL:DEFUN CL:DIRECTORY-NAMESTRING (PATHNAME)
  ;; Returns the directory part of PATHNAME as a string.
  (%%COMPONENT-STRING (%%PATHNAME-DIRECTORY (PATHNAME PATHNAME))))

(DECLARE%: DONTVAL@LOAD DOCOPY

(%%INITIALIZE-DEFAULT-PATHNAME)
)

(DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILEVARs

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA CL:ENOUGH-NAMESTRING CL:MERGE-PATHNAMES CL:MAKE-PATHNAME %%PRINT-PATHNAME)
)

(RPAQQ CMLPATHNAMECOMS
  ;; Common Lisp pathname functions
  (PROP FILETYPE CMLPATHNAME)
  (COMS ;; useful macros
    (FUNCTIONS %%WILD-NAME %%COMPONENT-STRING))
  (STRUCTURES PATHNAME DIRECTORY-COMPONENT)
  (FNS %%PRINT-PATHNAME CL:MAKE-PATHNAME %%PRINT-DIRECTORY-COMPONENT)
  (FUNCTIONS CL:PATHNAME-HOST CL:PATHNAME-DEVICE CL:PATHNAME-DIRECTORY CL:PATHNAME-NAME CL:PATHNAME-TYPE
    CL:PATHNAME-VERSION)
  (FNS PATHNAME CL:MERGE-PATHNAMES FILE-NAME CL:HOST-NAMESTRING CL:ENOUGH-NAMESTRING %%NUMERIC-STRING-P)
  (FUNCTIONS CL:NAMESTRING CL:PARSE-NAMESTRING CL:TRUENAME)
  (FUNCTIONS %%MAKE-PATHNAME)
  (FUNCTIONS %%PATHNAME-EQUAL %%DIRECTORY-COMPONENT-EQUAL)
  (FUNCTIONS %%INITIALIZE-DEFAULT-PATHNAME)
  (VARIABLES *DEFAULT-PATHNAME-DEFAULTS*)
  (COMS ;; Interlisp-D compatibility
    (FUNCTIONS INTERLISP-NAMESTRING UNPACKPATHNAME.STRING))
  (FUNCTIONS CL:FILE-NAMESTRING CL:DIRECTORY-NAMESTRING)
  (DECLARE%: DONTVAL@LOAD DOCOPY (P (%%INITIALIZE-DEFAULT-PATHNAME)))
  (DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILEVARs
    (ADDVARS (NLAMA)
              (NLAML)
              (LAMA CL:ENOUGH-NAMESTRING CL:HOST-NAMESTRING FILE-NAME CL:MERGE-PATHNAMES PATHNAME
                %%PRINT-DIRECTORY-COMPONENT CL:MAKE-PATHNAME %%PRINT-PATHNAME]))

(DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILEVARs

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA CL:ENOUGH-NAMESTRING CL:HOST-NAMESTRING FILE-NAME CL:MERGE-PATHNAMES PATHNAME
  %%PRINT-DIRECTORY-COMPONENT CL:MAKE-PATHNAME %%PRINT-PATHNAME)
)

```

FUNCTION INDEX

%%DIRECTORY-COMPONENT-EQUAL	7	FILE-NAME	4	CL:PATHNAME-DEVICE	3
%%INITIALIZE-DEFAULT-PATHNAME	7	CL:FILE-NAMESTRING	8	CL:PATHNAME-DIRECTORY	3
%%MAKE-PATHNAME	6	CL:HOST-NAMESTRING	4	CL:PATHNAME-HOST	3
%%NUMERIC-STRING-P	5	INTERLISP-NAMESTRING	7	CL:PATHNAME-NAME	3
%%PATHNAME-EQUAL	6	CL:MAKE-PATHNAME	2	CL:PATHNAME-TYPE	3
%%PRINT-DIRECTORY-COMPONENT	2	CL:MERGE-PATHNAMES	3	CL:PATHNAME-VERSION	3
%%PRINT-PATHNAME	1	CL:NAMESTRING	5	CL:TRUENAME	6
CL:DIRECTORY-NAMESTRING	8	CL:PARSE-NAMESTRING	6	UNPACKPATHNAME.STRING	7
CL:ENOUGH-NAMESTRING	4	PATHNAME	3		

STRUCTURE INDEX

DIRECTORY-COMPONENT	1	PATHNAME	1
-------------------------------	---	--------------------	---

MACRO INDEX

%%COMPONENT-STRING	1	%%WILD-NAME	1
------------------------------	---	-----------------------	---

VARIABLE INDEX

DEFAULT-PATHNAME-DEFAULTS	7
---------------------------------------	---

PROPERTY INDEX

CMLPATHNAME	1
-----------------------	---
