File created: 25-Mar-86 13:22:34 **{LOGOS:AFB:SIP}<DOUG>LISP>DUMPER.;10**

changes to: (FNS DUMP.DUMP DUMP.DIRECTORY DUMP DUMP.LOG.FILENAME DUMP.NEW.FILENAME DUMP.DIRECTORIES)
(VARS DUMPERCOMS)

previous date: 24-Mar-86 21:40:40 {LOGOS:AFB:SIP}<DOUG>LISP>DUMPER.;7

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS


```
(RPAQQ DUMPERCOMS ((FNS DUMP DUMP.DUMP DUMP.NEW.FILENAME DUMP.DIRECTORIES DUMP.DIRECTORY
                        DUMP.GENERATE.NEWERTHAN DUMP.NVERSIONS)
                   (INITVARS (DUMP.IGNORE.DIRS '(FONTS CLEARINGHOUSE SYSTEMFILES DESKTOPS))
                          (DUMP.IGNORE.SPECS '(*.DCOM;* *.SYSOUT;*))
                          (DUMP.DIRECTORY.SEPARATOR "\"))))

(DEFINEQ

(DUMP
  [LAMBDA (HOST TO.DIRECTORY LOG.FILE NEWERTHAN NVERSIONS)       (* drc: "25-Mar-86 12:43")

          (* * Will dump all files on NS file server HOST which are newer than NEWERTHAN, a date string, with a maximum of
          NVERSIONS of a particular file being dumped to TO.DIRECTORY.
          If NEWERTHAN is NIL than all versions written since (GDATE 0) will be dumped.
          If NVERSIONS is NIL then all versions will be dumped. A log file will be written to a file w/ name specified by
          DUMP.LOG.FILENAME. Returns the log file name.)

    (DECLARE (GLOBALVARS DUMP.IGNORE.SPECS))
    (RESETLST
        (LET ((IDATE (if NEWERTHAN
                         then (IDATE NEWERTHAN)
                       else 0))
              (FILTERS (MAPCAR DUMP.IGNORE.SPECS (FUNCTION DIRECTORY.MATCH.SETUP)))
              LOG FILES)                                          (* do a little arg checking)
             (OR (FIXP IDATE)
                 (ERROR NEWERTHAN "ARG NOT A DATE STRING"))
             (OR (STRPOS ":" (MKSTRING HOST))
                 (ERROR HOST "NOT AN NS HOST"))
             (SETQ LOG (OPENSTREAM LOG.FILE 'OUTPUT 'NEW))
             (RESETSAVE NIL (LIST 'CLOSEF? LOG))
             (printout LOG "Dump of server " HOST " on " (DATE)
                    " to " TO.DIRECTORY "." T "Dumping " (OR NVERSIONS "all")
                    " versions of each file written since "
                    (GDATE IDATE)
                    "." T)
             (printout LOG "Not dumping files on directories :")
             (for DIR in DUMP.IGNORE.DIRS do (PRINTOUT LOG " " DIR))
             (printout LOG T "Not dumping files which match :")
             (for SPEC in DUMP.IGNORE.SPECS do (PRINTOUT LOG " " SPEC))
             (printout LOG T T)                                   (* actually do the dump.)
             (for DIR in (DUMP.DIRECTORIES HOST) bind FILES do (PRINTOUT T DIR)
                                                               (SETQ FILES (DUMP.DIRECTORY HOST DIR IDATE
                                                                                  NVERSIONS FILTERS LOG))
                                                               (printout T "(" (FLENGTH FILES)
                                                                      ")")
                                                               (DUMP.DUMP FILES TO.DIRECTORY LOG)
                                                               (PRINTOUT T "OK" T))
             (printout LOG T (DATE)
                    " Done with backups." T)
             (CLOSEF LOG)))])

(DUMP.DUMP
  [LAMBDA (FILES TO.DIR LOG)                                     (* drc: "25-Mar-86 13:02")

          (* * Will copy all files in FILES to TO.DIR, renaming files as specified by DUMP.NEW.FILENAME.
          Each file copied is recorded in LOG, a stream opened for output.)

    (for FILE in FILES as N from 1 bind VALUE NEWNAME do (SETQ N9]
  AME
  (DUMP.NEW.FILENAME FILE TO.DIR))

(SETQ
  VALUE (NLSETQ (COPYFILE FILE NEWNAME)))

(if
  (LISTP VALUE)
  then                                                           (* file dumped successfully)
  (PRINTOUT T (if (ZEROP (REMAINDER N 10))
```

```
                then N
               else "."))
  (printout LOG FILE T)
  else                                                    (* error occurred)
  (LET ((ERROR (ERRORN)))
       (PRINTOUT T (ERRORSTRING (CAR ERROR))
              " "
              (CADR ERROR)
              T FILE " Not dumped." T)
       (PRINTOUT LOG (ERRORSTRING (CAR ERROR))
              " "
              (CADR ERROR)
              T FILE " Not dumped." T)))

)

NIL

[DUMP.NEW.FILENAME (LAMBDA (FILE TO.DIR)                  (* drc: "25-Mar-86 12:00")

          (* * Replaces >'s in the DIRECTORY field of file with DUMP.DIRECTORY.SEPARATOR.)

                    (DECLARE (GLOBALVARS DUMP.DIRECTORY.SEPARATOR))
                    (LET ((FILEFIELDS (UNPACKFILENAME FILE)))
                         (PACKFILENAME 'NAME (CONCAT [CONCATLIST (DSUBST DUMP.DIRECTORY.SEPARATOR '>
                                                                 (UNPACK (LISTGET FILEFIELDS 'DIRECTORY]
                                                    DUMP.DIRECTORY.SEPARATOR
                                                    (LISTGET FILEFIELDS 'NAME))
                                  'EXTENSION
                                  (LISTGET FILEFIELDS 'EXTENSION)
                                  'VERSION NIL 'BODY TO.DIRECTORY]

[DUMP.DIRECTORIES (LAMBDA (HOST)                          (* drc: "25-Mar-86 12:05")

          (* * Returns a list of the names of all the top-level directories on NS host HOST except those on DUMP.IGNORE.DIRS)

                    (DECLARE (GLOBALVARS DUMP.IGNORE.DIRS))
                    (LET [(DIRS (MAPCAR (DIRECTORY (PACKFILENAME 'HOST HOST 'DIRECTORY '*))
                                        (FUNCTION (LAMBDA (SPEC)  (* DIRECTORY of {host:}<*> returns a list of {host:}<*>.;1)
                                                  (MKATOM (U-CASE (FILENAMEFIELD SPEC 'DIRECTORY]
                         (for DIR in DUMP.IGNORE.DIRS do (DREMOVE (MKATOM (U-CASE DIR))
                                                                  DIRS))
                         DIRS]

[DUMP.DIRECTORY (LAMBDA (HOST DIR IDATE NVERSIONS FILTERS LOG)   (* drc: "25-Mar-86 13:03")

          (* * Return all the files on DIR newer than IDATE, an IDATE, with no more than NVERSIONS of any particular file.
          NIL versions means all. Files which match a filter on FILTERS (generated from by mapping DIRECTORY.MATCH.SETUP
          over DUMP.IGNORE.SPECS) are removed.)

                  (LET* ((SPEC (PACKFILENAME 'HOST HOST 'DIRECTORY DIR 'BODY '*.*;*))
                         (VALUE (NLSETQ (DUMP.GENERATE.NEWERTHAN SPEC IDATE)))
                         (FILES (if (LISTP VALUE)
                                    then (CAR VALUE)
                                  else (LET ((ERROR (ERRORN)))
                                            (PRINTOUT T (ERRORSTRING (CAR ERROR))
                                                   " "
                                                   (CADR ERROR)
                                                   T SPEC " Not dumped." T)
                                            (PRINTOUT LOG (ERRORSTRING (CAR ERROR))
                                                   " "
                                                   (CADR ERROR)
                                                   T SPEC " Not dumped." T))
                                       NIL)))
                        (for FILE in (if NVERSIONS
                                         then (DUMP.NVERSIONS FILES NVERSIONS)
                                       else FILES)
                          when (for FILTER in FILTERS never (DIRECTORY.MATCH FILTER FILE)) collect FILE]

[DUMP.GENERATE.NEWERTHAN (LAMBDA (SPEC IDATE)                     (* drc: "21-Mar-86 22:05")
                           (RESETLST                              (* collect all the files in filespec SPEC newerthan IDATE)
                               (bind FILE (GEN _ (\GENERATEFILES SPEC '  (WRITEDATE)
                                                                      '
                                                                   (RESETLST
                                                                       SORT)))
                                  eachtime (SETQ FILE (\GENERATENEXTFILE GEN))
                                  when (GEQ (IDATE (\GENERATEFILEINFO GEN 'WRITEDATE))
                                            IDATE)
                                  collect FILE until (NOT FILE)))]

[DUMP.NVERSIONS (LAMBDA (FILES N)                                 (* drc: " 1-Jan-01 00:36")
                                                                  (* assumes FILES is sorted, with low versions first)
                  (DREVERSE (for TAIL on (DREVERSE FILES) bind FILE LASTFILE FILEFIELDS LASTFIELDS
                                                         (M _ 1)
                            eachtime                              (* Have to reverse list to get high versions first.)
                                  (SETQ LASTFILE FILE)
                                  (SETQ FILE (CAR TAIL))
```

```
                                    (SETQ LASTFIELDS FILEFIELDS)
                                    (SETQ FILEFIELDS (UNPACKFILENAME FILE))
                                                                  (* only collect the first N of a particular file)
                                    (if (AND (EQ (LISTGET FILEFIELDS 'NAME)
                                                 (LISTGET LASTFIELDS 'NAME))
                                             (EQ (LISTGET FILEFIELDS 'EXTENSION)
                                                 (LISTGET LASTFIELDS 'EXTENSION))
                                             (EQ (LISTGET FILEFIELDS 'DIRECTORY)
                                                 (LISTGET LASTFIELDS 'DIRECTORY)))
                                        then (SETQ M (ADD1 M))
                                      else (SETQ M 1))
                                when (LEQ M N) collect FILE]
```

```
NIL
```

```
(RPAQ? DUMP.IGNORE.DIRS ' (FONTS CLEARINGHOUSE SYSTEMFILES DESKTOPS))
```

```
(RPAQ? DUMP.IGNORE.SPECS ' (*.DCOM;* *.SYSOUT;*))
```

```
(RPAQ? DUMP.DIRECTORY.SEPARATOR "\")
```

```
(PUTPROPS DUMPER COPYRIGHT ("Speech Input Project, Univ. of Edinburgh" 1986 1901))
```

FUNCTION INDEX

VARIABLE INDEX