File created: 23-Feb-2024 23:14:08 **{WMEDLEY}<library>lafite>LAFITE-ABBREV.;1**

edit by: rmk

previous date: 11-Nov-88 19:37:06 {WMEDLEY}<library>lafite>LAFITEABBREV.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ LAFITE-ABBREVCOMS
       ((APPENDVARS (LAFITE.ABBREVS ("*@*:*" "*@*:*" :OUT)
                           ("*@*" "%"*%%*%":GV:Xerox" :IN)
                           ("*@*" "*%%*:GV:Xerox")
                           ("*@*.*" "*%%*:*:Xerox" :IN)
                           ("*.pa" "*:PA:Xerox")))
        (INITVARS (LAFITE.ABBREV.DIRECTIONS :BOTH)
               (LAFITE.ABBREV.MOVE.GAZE.RIGHT T)
               (LAFITE.ABBREV.TRACE))
        (FUNCTIONS SAFEUPPERCHARCODE)
        (FNS LAFITE.ABBREV LAFITE.ABBREV.MATCH)
        (ADVISE (NSNAME.TO.STRING :IN \NSMAIL.PRINT.NAMES)
               (PARSE.NSNAME :IN \NSMAIL.PARSE1))
        (GLOBALVARS LAFITE.ABBREVS LAFITE.ABBREV.DIRECTIONS LAFITE.ABBREV.MOVE.GAZE.RIGHT LAFITE.ABBREV.TRACE)))

(APPENDTOVAR LAFITE.ABBREVS ("*@*:*" "*@*:*" :OUT)
       ("*@*" "%"*%%*%":GV:Xerox" :IN)
       ("*@*" "*%%*:GV:Xerox")
       ("*@*.*" "*%%*:*:Xerox" :IN)
       ("*.pa" "*:PA:Xerox"))

(RPAQ? LAFITE.ABBREV.DIRECTIONS :BOTH)

(RPAQ? LAFITE.ABBREV.MOVE.GAZE.RIGHT T)

(RPAQ? LAFITE.ABBREV.TRACE )


(DEFMACRO SAFEUPPERCHARCODE (CODE)
   `(if (AND (NUMBERP ,CODE)
             (LEQ ,CODE 255))
        THEN (GETCASEARRAY UPPERCASEARRAY ,CODE)
      ELSE ,CODE))

(DEFINEQ

(LAFITE.ABBREV
  [LAMBDA (ADDRESS DIRECTION)                                   ; Edited 2-Aug-88 16:48 by Lennart

;;; Translate Lafite abbreviations to addresses or vice versa.  The DIRECTION should be either :IN, :OUT, or :BOTH, defaulting to :OUT if left
;;; unspecified.  It has to agree with the overall setting of LAFITE.ABBREV.DIRECTIONS for anything to happen.  LAFITE.ABBREVS, then, is a list of
;;; translations -- each of the form (abbrev address direction), where direction is optional and defaults to LAFITE.ABBREV.DIRECTIONS.  For
;;; compatibility with TEDIT.ABBREVS, we also support the simpler form (abbrev . address), which is interpreted the same as (abbrev address NIL).
;;; Both abbrevations and addresses in LAFITE.ABBREVS may contain wildcards -- read more about this in the documentation.

    (OR DIRECTION (SETQ DIRECTION :OUT))
    [if (AND LAFITE.ABBREV.DIRECTIONS (OR (EQ LAFITE.ABBREV.DIRECTIONS :BOTH)
                                          (EQ LAFITE.ABBREV.DIRECTIONS DIRECTION)))
        then (for TRAN in LAFITE.ABBREVS bind NEWADDRESS when (AND (LISTP TRAN)
                                                                   (OR (NLISTP (CDR TRAN))
                                                                       (NULL (CDDR TRAN))
                                                                       (NOT (LITATOM (CADDR TRAN)))
                                                                       (EQ (CADDR TRAN)
                                                                           :BOTH)
                                                                       (EQ (CADDR TRAN)
                                                                           DIRECTION)))
                  do (if [SETQ NEWADDRESS (LAFITE.ABBREV.MATCH (if (EQ DIRECTION :OUT)
                                                                   then (CAR TRAN)
                                                                 elseif (LISTP (CDR TRAN))
                                                                   then (CADR TRAN)
                                                                 else (CDR TRAN))
                                                               ADDRESS
                                                               (if (EQ DIRECTION :IN)
                                                                   then (CAR TRAN)
                                                                 elseif (LISTP (CDR TRAN))
                                                                   then (CADR TRAN)
                                                                 else (CDR TRAN]
                      then ;; Success, now make sure there's no percent after the atsign unless user has turned off flag.
                           (AND LAFITE.ABBREV.MOVE.GAZE.RIGHT (bind (P _ (STRPOSL (CHARCODE (@))
                                                                                  NEWADDRESS))
                                                                    Q while (AND P (SETQ Q (STRPOSL (CHARCODE
                                                                                                      (%%))
                                                                                                    NEWADDRESS P)))
                                                               do (RPLCHARCODE NEWADDRESS P (CHARCODE %%))
                                                                  (RPLCHARCODE NEWADDRESS Q (CHARCODE @))
```

```
                                                                    (SETQ P Q)))
                            (AND LAFITE.ABBREV.TRACE (PRINTOUT LAFITE.ABBREV.TRACE T "[Translating " ADDRESS " to
" NEWADDRESS "]"))
                            (SETQ ADDRESS NEWADDRESS)
                            (RETURN]
    ADDRESS])
```

(LAFITE.ABBREV.MATCH

```
  [LAMBDA (PATTERN STRING TEMPLATE)                    ; Edited 22-Sep-88 13:03 by Lennart
    (DECLARE (LOCALVARS . T))
```

;;; Match a test string wrt to a pattern string where asterisks in the pattern act as wildcards, matching zero or more arbitrary characters in the test string.
;;; The returned value is NIL or T, or if a template is given, a string merge done by substituting asterisks in the template string for the matched substrings
;;; in the test string. Be aware that this functions utilizes the fact that NTHCHARCODE will return NIL for chars outside of the string.

```
    (PROG ((patLen (NCHARS PATTERN))
           (strLen (NCHARS STRING))
           (temLen (NCHARS TEMPLATE))
           (pp 1)
           (sp 1)
           (tp 1)
           pwp swp twp stack matches)
      LOOP
          (SETQ pwp pp)
          (SETQ swp sp)
          (bind tmp eachtime (SETQ tmp (NTHCHARCODE PATTERN pwp))
                             (if (EQ tmp (CHARCODE *))
                                 then (RETURN))
             while (EQ (SAFEUPPERCHARCODE tmp)
                       (SAFEUPPERCHARCODE (NTHCHARCODE STRING swp)))
             do (if (IGREATERP pwp patLen)
                    then (GO SUCCEED))
                (SETQ pwp (ADD1 pwp))
                (SETQ swp (ADD1 swp))
             finally (GO BACKTRACK))
          (SETQ pp (ADD1 pwp))
          (SETQ sp swp)
      EXTEND
          (bind (tmp _ (SAFEUPPERCHARCODE (NTHCHARCODE PATTERN pp)))
             while (NEQ tmp (SAFEUPPERCHARCODE (NTHCHARCODE STRING sp)))
             do (if (IGREATERP sp strLen)
                    then (GO BACKTRACK))
                (SETQ sp (ADD1 sp)))
          (push stack pp swp sp)
          (GO LOOP)
      BACKTRACK
          (if (NULL stack)
              then (RETURN NIL))
          (SETQ pp (pop stack))
          (SETQ swp (pop stack))
          (SETQ sp (ADD1 (pop stack)))
          (GO EXTEND)
      SUCCEED
          (if (NOT TEMPLATE)
              then (RETURN (OR STRING T)))
      ;; Collect the matched substrings.
          [while stack do (SETQ pp (pop stack))
                          (SETQ swp (pop stack))
                          (SETQ sp (pop stack))
                          (push matches (SUBSTRING STRING swp (SUB1 sp]
      ;; Substitute wildcards in the template string.
          [RETURN (CONCATLIST (while (LEQ tp temLen) bind result eachtime (SETQ twp (OR (STRPOS "*" TEMPLATE tp)
                                                                                        (ADD1 temLen)))
                                                     do                    ; Note that (MKLIST NIL) is NIL
                                                        [SETQ result (NCONC result (MKLIST (SUBSTRING TEMPLATE tp (SUB1 twp)))
                                                                           (AND (ILEQ twp temLen)
                                                                                (MKLIST (pop matches]
                                                        (SETQ tp (ADD1 twp))
                                                     finally (RETURN result]
      FAIL
          (RETURN NIL])

)

[XCL:REINSTALL-ADVICE '(NSNAME.TO.STRING :IN \NSMAIL.PRINT.NAMES)
       :AFTER
       '((:LAST (SETQ !VALUE (LAFITE.ABBREV !VALUE :IN]

[XCL:REINSTALL-ADVICE '(PARSE.NSNAME :IN \NSMAIL.PARSE1)
       :BEFORE
       '((:LAST (SETQ NAME (LAFITE.ABBREV NAME :OUT]

(READVISE (NSNAME.TO.STRING :IN \NSMAIL.PRINT.NAMES)
       (PARSE.NSNAME :IN \NSMAIL.PARSE1))
```

```
(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS LAFITE.ABBREVS LAFITE.ABBREV.DIRECTIONS LAFITE.ABBREV.MOVE.GAZE.RIGHT LAFITE.ABBREV.TRACE)
)
```

FUNCTION INDEX

VARIABLE INDEX

ADVICE INDEX

MACRO INDEX