File created: 4-Jun-88 20:24:37 **{ERIS}<LISPCORE>LIBRARY>MSCOMMON.;4**

changes to: (TEMPLATES CL:PROGV)

previous date: 27-Apr-88 16:54:40 {ERIS}<LISPCORE>LIBRARY>MSCOMMON.;3

Read Table: XCL

Package: INTERLISP

Format: XCCS

```
(RPAQQ MSCOMMONCOMS
       ((PROP FILETYPE MSCOMMON)
        (FNS FUNCTIONSMSGETDEF FUNCTIONSMSMC VARIABLESMSGETDEF)

        ;; Templates for CL stuff that need them.

        (TEMPLATES ADD-EXEC CL:ADJOIN CL:ADJUST-ARRAY CL:APPLY CL:APPLYHOOK ASET CL:ASSOC CL:CLOSE CLRHASH
               CL:COMPILE CL:COMPILE-FILE CL:COMPILER-LET CL:COUNT CL:COUNT-IF CL:COUNT-IF-NOT CL:DECF DECLARE
               CL:DELETE CL:DELETE-DUPLICATES CL:DELETE-IF CL:DELETE-IF-NOT CL:EVAL-WHEN CL:EVALHOOK EXEC
               EXEC-EVAL CL:FILL FILL-VECTOR CL:FIND CL:FIND-IF CL:FIND-IF-NOT CL:FLET CL:FUNCTION CL:GETF
               CL:IN-PACKAGE CL:INCF CL:INTERSECTION CL:LABELS CL:LOAD CL:MACROLET CL:MAKE-ARRAY
               COMPILER:MAKE-CONTEXT CL:MAKE-HASH-TABLE CL:MAKE-LIST CL:MAKE-PACKAGE CL:MAKE-PATHNAME
               CL:MAKE-SEQUENCE CL:MAKE-STRING CL:MAPC CL:MAPCAN CL:MAPCAR CL:MAPCON CL:MAPHASH CL:MAPL
               CL:MAPLIST CL:MEMBER CL:MEMBER-IF CL:MEMBER-IF-NOT CL:MERGE CL:MISMATCH CL:MULTIPLE-VALUE-CALL
               CL:MULTIPLE-VALUE-PROG1 CL:MULTIPLE-VALUE-SETQ CL:NINTERSECTION CL:NRECONC CL:NREVERSE
               CL:NSET-DIFFERENCE CL:NSET-EXCLUSIVE-OR CL:NSTRING-CAPITALIZE CL:NSTRING-DOWNCASE
               CL:NSTRING-UPCASE CL:NSUBLIS CL:NSUBST CL:NSUBST-IF CL:NSUBST-IF-NOT CL:NSUBSTITUTE
               CL:NSUBSTITUTE-IF CL:NSUBSTITUTE-IF-NOT CL:NUNION OPEN CL:PARSE-INTEGER CL:PARSE-NAMESTRING
               CL:POP CL:POSITION CL:POSITION-IF CL:POSITION-IF-NOT CL:PROGV CL:PSETF CL:PSETQ CL:PUSH
               CL:PUSHNEW CL:RASSOC CL:READ-FROM-STRING CL:REDUCE CL:REMF CL:REMOVE CL:REMOVE-DUPLICATES
               CL:REMOVE-IF CL:REMOVE-IF-NOT CL:REPLACE CL:ROTATEF CL:SEARCH CL:SET-DIFFERENCE
               CL:SET-EXCLUSIVE-OR CL:SHIFTF CL:SORT CL:STABLE-SORT CL:STRING-CAPITALIZE CL:STRING-DOWNCASE
               STRING-EQUAL CL:STRING-GREATERP CL:STRING-LESSP CL:STRING-NOT-EQUAL CL:STRING-NOT-GREATERP
               CL:STRING-NOT-LESSP CL:STRING-UPCASE CL:STRING/= CL:STRING< CL:STRING<= CL:STRING= CL:STRING>
               CL:STRING>= CL:SUBLIS CL:SUBSETP CL:SUBST CL:SUBST-IF CL:SUBST-IF-NOT CL:SUBSTITUTE
               CL:SUBSTITUTE-IF CL:SUBSTITUTE-IF-NOT CL:TREE-EQUAL CL:UNION CL:VECTOR-PUSH CL:VECTOR-PUSH-EXTEND
               WRITE CL:WRITE-LINE CL:WRITE-STRING CL:WRITE-TO-STRING)
        (P ;; First tell Masterscope how to find FUNCTIONS and VARIABLES

           (MSADDANALYZE 'VARIABLES 'VARIABLE 'VARIABLES 'VARIABLESMSGETDEF)
           (MSADDANALYZE 'FUNCTIONS 'FUNCTION 'FUNCTIONS 'FUNCTIONSMSGETDEF 'FUNCTIONSMSMC)

           ;; Then add KEYWORD support.  Templates may now contain the following as their last element:

           ;; ... KEYWORDS list of keywords accepted)

           ;; No (list of keywords accepted) means use keywords gathered from analyzed source.  This must naturally be last in a template.

           (MSADDRELATION '(ACCEPT ACCEPTS ACCEPTING ACCEPTED)
                  '(KEYACCEPT))
           (MSADDRELATION '(SPECIFY SPECIFIES SPECIFYING SPECIFIED)
                  '(KEYSPECIFY))
           (MSADDRELATION '(KEYCALL KEYCALLS KEYCALLING KEYCALLED))
           (MSADDMODIFIER 'ACCEPT 'KEYWORD 'KEYACCEPT)
           (MSADDMODIFIER 'ACCEPT 'KEYWORDS 'KEYACCEPT)
           (MSADDMODIFIER 'SPECIFY 'KEYWORD 'KEYSPECIFY)
           (MSADDMODIFIER 'SPECIFY 'KEYWORDS 'KEYSPECIFY)

           ;; Stuff for locally-defined things.  We don't attempt to handle them (*sigh*), just record them.

           (MSADDRELATION '(FLET FLETS FLETTING FLET))
           (MSADDRELATION '(LABEL LABELS LABELLING LABELLED))
           (MSADDRELATION '(MACROLET MACROLETS MACROLETTING MACROLET))
           (MSADDRELATION '(LOCAL-DEFINE LOCAL-DEFINES LOCAL-DEFINING LOCAL-DEFINED)
                  '(FLET LABEL MACROLET))

           ;; What the heck, track COMPILER-LETs.

           (MSADDRELATION '(COMPILER-LET COMPILER-LETS COMPILER-LETTING COMPILER-LETTED))

           ;; Finally, copy the templates over into MSTEMPLATES and clear the USERTEMPLATES table now; no need for the Common Lisp
           ;; templates to live there.

           (MAPHASH USERTEMPLATES #'(LAMBDA (VAL KEY)
                                           (PUTHASH KEY VAL MSTEMPLATES)))
           (CLRHASH USERTEMPLATES))))

(PUTPROPS MSCOMMON FILETYPE :COMPILE-FILE)

(DEFINEQ

(FUNCTIONSMSGETDEF
  (LAMBDA (NAME TYPE SOURCE OPTIONS)                                   ; Edited 31-Mar-88 17:31 by jrb:
    (LET ((BODY (REMOVE-COMMENTS (GETDEF NAME 'FUNCTIONS SOURCE OPTIONS))))
         (AND BODY (SELECTQ (CAR BODY)
                       (DEFMACRO (OR (GETTEMPLATE NAME)
                                     (SETTEMPLATE NAME 'MACRO))
                                 NIL)
                       (CL:DEFUN ;; Body is of the form:
```

```
                                 ;; (DEFUN name (args...) bodies...)
                                 ;; We want to hand Masterscope a massaged form it will understand.
                                 ;; Which I believe is of this form:
                                 `(CL:LAMBDA ,(CADDR BODY)
                                         ,@(CDDDR BODY)))
                       NIL)))))

(FUNCTIONSMSMC
  (LAMBDA (NAME TYPE REASON)                                    ; Edited  1-Apr-88 13:47 by jrb:
    ;; Trick here is we don't want to mark FUNCTIONS macros as changed because they really don't get analyzed, but we do want to call
    ;; CHANGEMACRO for them
    (|if| (EQ (CAR (GETDEF NAME 'FUNCTIONS NIL '(NOERROR)))
            'DEFMACRO)
        |then| (CHANGEMACRO NAME TYPE REASON)
             NIL
      |else| T)))

(VARIABLESMSGETDEF
  (LAMBDA (NAME TYPE SOURCE OPTIONS)                            ; Edited 19-Feb-88 19:46 by jrb:
    (LET ((BODY (GETDEF NAME 'VARIABLES SOURCE OPTIONS))
          SPECVARP)
         (AND BODY
              ;; We have to return something here so Masterscope can get hold of the init form, and so It'll stop looking for other things
              `(CL:LAMBDA NIL ,(IF (CADDR BODY)
                                   THEN `(SETQ ,(CADR BODY)
                                          ,(CADDR BODY)))))))))
)

;; Templates for CL stuff that need them.

(SETTEMPLATE 'ADD-EXEC '(KEYWORDS :PROFILE :REGION :TTY :EXEC :ID))

(SETTEMPLATE 'CL:ADJOIN '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:ADJUST-ARRAY '(SMASH EVAL KEYWORDS :ELEMENT-TYPE :INITIAL-ELEMENT :INITIAL-CONTENTS
                                      :FILL-POINTER :DISPLACED-TO :DISPLACED-INDEX-OFFSET :FATP
                                      :DISPLACED-TO-BASE))

(SETTEMPLATE 'CL:APPLY '((! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                         |then| (LIST 'SET 'EVAL)
                                       |else| (LIST 'SMASH 'EVAL)))))

(SETTEMPLATE 'CL:APPLYHOOK '((! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                             |then| (LIST 'SET 'EVAL)
                                           |else| (LIST 'SMASH 'EVAL)))))

(SETTEMPLATE 'ASET '(EVAL SMASH |..| EVAL))

(SETTEMPLATE 'CL:ASSOC '(EVAL EVAL KEYWORDS :TEST :TEST-NOT))

(SETTEMPLATE 'CL:CLOSE '(EVAL KEYWORDS :ABORT))

(SETTEMPLATE 'CLRHASH '((! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                        |then| (LIST 'SET 'EVAL)
                                      |else| (LIST 'SMASH 'EVAL)))))

(SETTEMPLATE 'CL:COMPILE '(EVAL EVAL KEYWORDS :LAP))

(SETTEMPLATE 'CL:COMPILE-FILE '(EVAL KEYWORDS :OUTPUT-FILE :ERROR-FILE :ERRORS-TO-TERMINAL :LAP-FILE :LOAD
                                     :FILE-MANAGER-FORMAT :PROCESS-ENTIRE-FILE))

(SETTEMPLATE 'CL:COMPILER-LET '(! NIL (BOTH (|..| (IF LISTP (NIL EVAL |..| EFFECT)
                                                     NIL))
                                            (|..| (IF LISTP ((BOTH BIND COMPILER-LET))
                                                     (BOTH BIND COMPILER-LET))))
                                  |..| EFFECT RETURN))

(SETTEMPLATE 'CL:COUNT '(EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :KEY))

(SETTEMPLATE 'CL:COUNT-IF '(EVAL EVAL KEYWORDS :FROM-END :START :END :KEY))

(SETTEMPLATE 'CL:COUNT-IF-NOT '(EVAL EVAL KEYWORDS :FROM-END :START :END :KEY))

(SETTEMPLATE 'CL:DECF '(! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                       |then| (LIST 'SET 'EVAL)
                                     |else| (LIST 'SMASH 'EVAL))))

(SETTEMPLATE 'DECLARE '(|..| (@ EXPR (CONS NIL (SELECTQ (CAR (LISTP EXPR))
                                                   (LOCALVARS '(IF LISTP (|..| LOCALVARS)
                                                                  LOCALVARS))
                                                   ((SPECVARS CL:SPECIAL)
                                                        '(IF LISTP (|..| SPECVARS)
                                                            SPECVARS))
                                                   NIL)))))

(SETTEMPLATE 'CL:DELETE '(EVAL SMASH KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:DELETE-DUPLICATES '(SMASH KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :KEY))

(SETTEMPLATE 'CL:DELETE-IF '(CL:FUNCTION EVAL KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:DELETE-IF-NOT '(CL:FUNCTION EVAL KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:EVAL-WHEN '(NIL |..| EFFECT RETURN))

(SETTEMPLATE 'CL:EVALHOOK '((! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                            |then| (LIST 'SET 'EVAL)
                                          |else| (LIST 'SMASH 'EVAL)))))

(SETTEMPLATE 'EXEC '(KEYWORDS :TOP-LEVEL-P :WINDOW :TITLE :COMMAND-TABLES :ENVIRONMENT :PROMPT :FUNCTION
                           :PROFILE :ID))

(SETTEMPLATE 'EXEC-EVAL '(EVAL EVAL KEYWORDS :PROMPT :ID :TYPE))

(SETTEMPLATE 'CL:FILL '(SMASH EVAL KEYWORDS :START :END))

(SETTEMPLATE 'FILL-VECTOR '(SMASH EVAL KEYWORDS :START :END))

(SETTEMPLATE 'CL:FIND '(EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :KEY))

(SETTEMPLATE 'CL:FIND-IF '(EVAL EVAL KEYWORDS :FROM-END :START :END :KEY))

(SETTEMPLATE 'CL:FIND-IF-NOT '(EVAL EVAL KEYWORDS :FROM-END :START :END :KEY))

(SETTEMPLATE 'CL:FLET '((|..| (FLET))
                        |..| EFFECT RETURN))

(SETTEMPLATE 'CL:FUNCTION '((REMOTE (IF LITATOM CALL LAMBDA))
                            (IF LITATOM EVAL NIL)))

(SETTEMPLATE 'CL:GETF '(EVAL PROP EVAL))

(SETTEMPLATE 'CL:IN-PACKAGE '(EVAL KEYWORDS :NICKNAMES :USE))

(SETTEMPLATE 'CL:INCF '(! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                       |then| (LIST 'SET 'EVAL)
                                     |else| (LIST 'SMASH 'EVAL))))

(SETTEMPLATE 'CL:INTERSECTION '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:LABELS '((|..| (LABEL))
                          |..| EFFECT RETURN))

(SETTEMPLATE 'CL:LOAD '(EVAL KEYWORDS :VERBOSE :PRINT :IF-DOES-NOT-EXIST :PACKAGE :LOADFLG))

(SETTEMPLATE 'CL:MACROLET '((|..| (MACROLET))
                            |..| EFFECT RETURN))

(SETTEMPLATE 'CL:MAKE-ARRAY '(EVAL KEYWORDS :ELEMENT-TYPE :INITIAL-ELEMENT :INITIAL-CONTENTS :ADJUSTABLE
                                   :FILL-POINTER :DISPLACED-TO :DISPLACED-INDEX-OFFSET :FATP :EXTENDABLE
                                   :READ-ONLY-P :DISPLACED-TO-BASE))

(SETTEMPLATE 'COMPILER:MAKE-CONTEXT '(KEYWORDS :TOP-LEVEL-P :VALUES-USED :PREDICATE-P))

(SETTEMPLATE 'CL:MAKE-HASH-TABLE '(KEYWORDS :TEST :SIZE :REHASH-SIZE :REHASH-THRESHOLD))

(SETTEMPLATE 'CL:MAKE-LIST '(EVAL KEYWORDS :INITIAL-ELEMENT))

(SETTEMPLATE 'CL:MAKE-PACKAGE '(EVAL KEYWORDS :NICKNAMES :USE :PREFIX-NAME :INTERNAL-SYMBOLS :EXTERNAL-SYMBOLS
                                     :EXTERNAL-ONLY))

(SETTEMPLATE 'CL:MAKE-PATHNAME '(KEYWORDS :HOST :DEVICE :DIRECTORY :NAME :TYPE :VERSION :DEFAULTS))

(SETTEMPLATE 'CL:MAKE-SEQUENCE '(EVAL EVAL KEYWORDS :INITIAL-ELEMENT))

(SETTEMPLATE 'CL:MAKE-STRING '(EVAL KEYWORDS :INITIAL-ELEMENT))

(SETTEMPLATE 'CL:MAPC '(FUNCTION |..| EVAL))

(SETTEMPLATE 'CL:MAPCAN '(FUNCTION |..| EVAL))

(SETTEMPLATE 'CL:MAPCAR '(FUNCTION |..| EVAL))

(SETTEMPLATE 'CL:MAPCON '(FUNCTION |..| EVAL))

(SETTEMPLATE 'CL:MAPHASH '(FUNCTION EVAL))
```

```
(SETTEMPLATE ’CL:MAPL ’(FUNCTION |..| EVAL))

(SETTEMPLATE ’CL:MAPLIST ’(FUNCTION |..| EVAL))

(SETTEMPLATE ’CL:MEMBER ’(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’CL:MEMBER-IF ’(EVAL EVAL KEYWORDS :KEY))

(SETTEMPLATE ’CL:MEMBER-IF-NOT ’(EVAL EVAL KEYWORDS :KEY))

(SETTEMPLATE ’CL:MERGE ’(EVAL EVAL EVAL EVAL KEYWORDS :KEY))

(SETTEMPLATE ’CL:MISMATCH ’(EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :KEY :START1 :START2 :END1 :END2))

(SETTEMPLATE ’CL:MULTIPLE-VALUE-CALL ’(FUNCTION |..| EVAL))

(SETTEMPLATE ’CL:MULTIPLE-VALUE-PROG1 ’(RETURN |..| EFFECT))

(SETTEMPLATE ’CL:MULTIPLE-VALUE-SETQ ’((|..| SET)
                                       EVAL))

(SETTEMPLATE ’CL:NINTERSECTION ’(SMASH EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’CL:NRECONC ’((! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                           |then| (LIST ’SET ’EVAL)
                                         |else| (LIST ’SMASH ’EVAL)))))

(SETTEMPLATE ’CL:NREVERSE ’((! NIL EXPR (|if| (LITATOM (CAR EXPR))
                                            |then| (LIST ’SET ’EVAL)
                                          |else| (LIST ’SMASH ’EVAL)))))

(SETTEMPLATE ’CL:NSET-DIFFERENCE ’(SMASH EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’CL:NSET-EXCLUSIVE-OR ’(SMASH SMASH KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’CL:NSTRING-CAPITALIZE ’(SMASH KEYWORDS :START :END))

(SETTEMPLATE ’CL:NSTRING-DOWNCASE ’(SMASH KEYWORDS :START :END))

(SETTEMPLATE ’CL:NSTRING-UPCASE ’(SMASH KEYWORDS :START :END))

(SETTEMPLATE ’CL:NSUBLIS ’(EVAL SMASH KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’CL:NSUBST ’(EVAL EVAL SMASH KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’CL:NSUBST-IF ’(EVAL FUNCTION SMASH KEYWORDS :KEY))

(SETTEMPLATE ’CL:NSUBST-IF-NOT ’(EVAL FUNCTION SMASH KEYWORDS :KEY))

(SETTEMPLATE ’CL:NSUBSTITUTE ’(EVAL EVAL SMASH KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :COUNT :KEY))

(SETTEMPLATE ’CL:NSUBSTITUTE-IF ’(EVAL FUNCTION SMASH KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE ’CL:NSUBSTITUTE-IF-NOT ’(EVAL FUNCTION SMASH KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE ’CL:NUNION ’(SMASH SMASH KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE ’OPEN ’(EVAL KEYWORDS :DIRECTION :ELEMENT-TYPE :IF-EXISTS :IF-DOES-NOT-EXIST))

(SETTEMPLATE ’CL:PARSE-INTEGER ’(EVAL KEYWORDS :START :END :RADIX :JUNK-ALLOWED))

(SETTEMPLATE ’CL:PARSE-NAMESTRING ’(EVAL EVAL EVAL KEYWORDS :START :END :JUNK-ALLOWED))

(SETTEMPLATE ’CL:POP ’(! NIL EXPR (IF (ATOM (CAR EXPR))
                                      THEN ’(SET)
                                    ELSE ’(SMASH))))

(SETTEMPLATE ’CL:POSITION ’(EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :KEY))

(SETTEMPLATE ’CL:POSITION-IF ’(EVAL EVAL KEYWORDS :FROM-END :START :END :KEY))

(SETTEMPLATE ’CL:POSITION-IF-NOT ’(EVAL EVAL KEYWORDS :FROM-END :START :END :KEY))

(SETTEMPLATE ’CL:PROGV ’(EVAL EVAL |..| EFFECT RETURN))

(SETTEMPLATE ’CL:PSETF ’(! @ EXPR (CONS NIL (MAPCON (CDR EXPR)
                                                    (FUNCTION (LAMBDA (X)
                                                                (|if| (LITATOM (CAR X))
                                                                    |then| (LIST ’SET ’EVAL)
                                                                  |else| (LIST ’SMASH ’EVAL))))
                                                    (FUNCTION (LAMBDA (X)
                                                                (CDDR X)))))))

(SETTEMPLATE ’CL:PSETQ ’(! @ EXPR (CONS NIL (MAPCON (CDR EXPR)
                                                    (FUNCTION (LAMBDA (X)
                                                                (|if| (LITATOM (CAR X))
                                                                    |then| (LIST ’SET ’EVAL)
```

```
                                                                 |else| (LIST 'SMASH 'EVAL))))
                                                   (FUNCTION (LAMBDA (X)
                                                               (CDDR X)))))))

(SETTEMPLATE 'CL:PUSH '(! NIL @ EXPR (IF (ATOM (CADDR EXPR))
                                         THEN '(EVAL SET)
                                       ELSE '(EVAL SMASH))))

(SETTEMPLATE 'CL:PUSHNEW '(@ EXPR `(EVAL ,(IF (ATOM (CADDR EXPR))
                                              THEN 'SET
                                            ELSE 'SMASH)
                                         KEYWORDS :TEST :TEST-NOT :KEY)))

(SETTEMPLATE 'CL:RASSOC '(EVAL EVAL KEYWORDS :TEST :TEST-NOT))

(SETTEMPLATE 'CL:READ-FROM-STRING '(EVAL EVAL EVAL KEYWORDS :START :END :PRESERVE-WHITESPACE))

(SETTEMPLATE 'CL:REDUCE '(FUNCTION EVAL KEYWORDS :FROM-END :START :END :INITIAL-VALUE))

(SETTEMPLATE 'CL:REMF '(@ EXPR (IF (ATOM (CAR EXPR))
                                   THEN '(SET PROP)
                                 ELSE '(SMASH PROP))))

(SETTEMPLATE 'CL:REMOVE '(EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:REMOVE-DUPLICATES '(EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :KEY))

(SETTEMPLATE 'CL:REMOVE-IF '(EVAL EVAL KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:REMOVE-IF-NOT '(EVAL EVAL KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:REPLACE '(SMASH EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:ROTATEF '(|..| (IF (ATOM EXPR)
                                    SET SMASH)))

(SETTEMPLATE 'CL:SEARCH '(EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :KEY :START1 :START2 :END1 :END2))

(SETTEMPLATE 'CL:SET-DIFFERENCE '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:SET-EXCLUSIVE-OR '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:SHIFTF '(|..| (IF (ATOM EXPR)
                                   SET SMASH)
                               EVAL))

(SETTEMPLATE 'CL:SORT '(EVAL FUNCTION KEYWORDS :KEY))

(SETTEMPLATE 'CL:STABLE-SORT '(EVAL FUNCTION KEYWORDS :KEY))

(SETTEMPLATE 'CL:STRING-CAPITALIZE '(EVAL KEYWORDS :START :END))

(SETTEMPLATE 'CL:STRING-DOWNCASE '(EVAL KEYWORDS :START :END))

(SETTEMPLATE 'STRING-EQUAL '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING-GREATERP '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING-LESSP '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING-NOT-EQUAL '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING-NOT-GREATERP '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING-NOT-LESSP '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING-UPCASE '(EVAL KEYWORDS :START :END))

(SETTEMPLATE 'CL:STRING/= '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING< '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING<= '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING= '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING> '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:STRING>= '(EVAL EVAL KEYWORDS :START1 :END1 :START2 :END2))

(SETTEMPLATE 'CL:SUBLIS '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:SUBSETP '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:SUBST '(EVAL EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:SUBST-IF '(EVAL EVAL EVAL KEYWORDS :KEY))
```

```
(SETTEMPLATE 'CL:SUBST-IF-NOT '(EVAL EVAL EVAL KEYWORDS :KEY))

(SETTEMPLATE 'CL:SUBSTITUTE '(EVAL EVAL EVAL KEYWORDS :FROM-END :TEST :TEST-NOT :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:SUBSTITUTE-IF '(EVAL EVAL EVAL KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:SUBSTITUTE-IF-NOT '(EVAL EVAL EVAL KEYWORDS :FROM-END :START :END :COUNT :KEY))

(SETTEMPLATE 'CL:TREE-EQUAL '(EVAL EVAL KEYWORDS :TEST :TEST-NOT))

(SETTEMPLATE 'CL:UNION '(EVAL EVAL KEYWORDS :TEST :TEST-NOT :KEY))

(SETTEMPLATE 'CL:VECTOR-PUSH '(EVAL SMASH))

(SETTEMPLATE 'CL:VECTOR-PUSH-EXTEND '(EVAL SMASH EVAL))

(SETTEMPLATE 'WRITE '(EVAL KEYWORDS :STREAM :ESCAPE :RADIX :BASE :CIRCLE :PRETTY :LEVEL :LENGTH :CASE :GENSYM
                           :ARRAY))

(SETTEMPLATE 'CL:WRITE-LINE '(EVAL EVAL KEYWORDS :START :END))

(SETTEMPLATE 'CL:WRITE-STRING '(EVAL EVAL KEYWORDS :START :END))

(SETTEMPLATE 'CL:WRITE-TO-STRING '(EVAL KEYWORDS :ESCAPE :RADIX :BASE :CIRCLE :PRETTY :LEVEL :LENGTH :CASE
                                        :GENSYM :ARRAY))

;; First tell Masterscope how to find FUNCTIONS and VARIABLES

(MSADDANALYZE 'VARIABLES 'VARIABLE 'VARIABLES 'VARIABLESMSGETDEF)

(MSADDANALYZE 'FUNCTIONS 'FUNCTION 'FUNCTIONS 'FUNCTIONSMSGETDEF 'FUNCTIONSMSMC)

;; Then add KEYWORD support.  Templates may now contain the following as their last element:
;; ... KEYWORDS list of keywords accepted)
;; No (list of keywords accepted) means use keywords gathered from analyzed source.  This must naturally be last in a template.

(MSADDRELATION '(ACCEPT ACCEPTS ACCEPTING ACCEPTED)
       '(KEYACCEPT))

(MSADDRELATION '(SPECIFY SPECIFIES SPECIFYING SPECIFIED)
       '(KEYSPECIFY))

(MSADDRELATION '(KEYCALL KEYCALLS KEYCALLING KEYCALLED))

(MSADDMODIFIER 'ACCEPT 'KEYWORD 'KEYACCEPT)

(MSADDMODIFIER 'ACCEPT 'KEYWORDS 'KEYACCEPT)

(MSADDMODIFIER 'SPECIFY 'KEYWORD 'KEYSPECIFY)

(MSADDMODIFIER 'SPECIFY 'KEYWORDS 'KEYSPECIFY)

;; Stuff for locally-defined things.  We don't attempt to handle them (*sigh*), just record them.

(MSADDRELATION '(FLET FLETS FLETTING FLET))

(MSADDRELATION '(LABEL LABELS LABELLING LABELLED))

(MSADDRELATION '(MACROLET MACROLETS MACROLETTING MACROLET))

(MSADDRELATION '(LOCAL-DEFINE LOCAL-DEFINES LOCAL-DEFINING LOCAL-DEFINED)
       '(FLET LABEL MACROLET))

;; What the heck, track COMPILER-LETs.

(MSADDRELATION '(COMPILER-LET COMPILER-LETS COMPILER-LETTING COMPILER-LETTED))

;; Finally, copy the templates over into MSTEMPLATES and clear the USERTEMPLATES table now; no need for the Common Lisp templates to live
;; there.

(MAPHASH USERTEMPLATES #'(LAMBDA (VAL KEY)
                           (PUTHASH KEY VAL MSTEMPLATES)))

(CLRHASH USERTEMPLATES)

(PUTPROPS MSCOMMON COPYRIGHT ("Xerox Corporation" 1988))
```

{MEDLEY}<loops>obsolete>MSCOMMON.;1 30-Jun-2024 08:54:28
-- Listed on 30-Jun-2024 09:23:27 --

FUNCTION INDEX

TEMPLATE INDEX

PROPERTY INDEX