

File created: 4-Jun-88 20:04:55 {ERIS}<LISPCORE>LIBRARY>MSANALYZE.;10

changes to: (FNS MSPRGLAMBDA)  
(VARS MSANALYZECOMS)

previous date: 21-Apr-88 17:36:26 {ERIS}<LISPCORE>LIBRARY>MSANALYZE.;9

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
::  
:: Copyright (c) 1982, 1983, 1984, 1985, 1986, 1988 by Xerox Corporation. All rights reserved.  
:: The following program was created in 1982 but has not been published  
:: within the meaning of the copyright law, is furnished under license,  
:: and may not be used, copied and/or disclosed except in accordance  
:: with the terms of said license.  
::
```

### (RPAQQ MSANALYZECOMS

```
[ (PROP FILETYPE MSANALYZE)  
  (COMS (FNS VARS FREEVARS CALLS COLLECTFNDDATA CALLS3)  
    (VARS MSMACROPROPS (NOPACKCALLSFLG))  
    (BLOCKS (CALLS CALLS COLLECTFNDDATA CALLS3 (LOCALFREEVARS CALLSDATA)  
      (NOLINKFNS . T)  
      (GLOBALVARS INVISIBLEVARS COMPILEUSERFN NOPACKCALLSFLG LAMBDA$PLST MSRECORDTRANFLG)  
    )  
    (NIL VARS FREEVARS (LOCALVARS . T)))  
  (DECLARE%: EVAL@COMPILE (VARS MS.VERB.TO.NOTICED)  
    DONTCOPY  
    (MACROS MSVBNOTICED)))  
[COMS (FNS ALLCALLS MSINITFNDDATA MSPRGE MSPRGMACRO MSPRGCALL MSBINDVAR MSPRGRECORD MSPRGERR  
  MSPRGTEMPLATE1 MSPRGTEMPLATE MSPRGLAMBDA MSPRGLST ADDTO NLAMBDAFNP MSPRGDWIM MSDWIMTRAN)  
  (E (MAPC MSFNDDATA (FUNCTION REPLACD)))  
  (VARS MSFNDDATA MSERRORFN (MSRECORDTRANFLG))  
  (ADDVARS (INVISIBLEVARS $$1 $$2 $$3 $$4 $$5 $$6 $$7 $$8 $$9 $$10 $$11 $$12 $$13 $$14 $$15 $$16  
    $$17))  
  (DECLARE%: DONTCOPY (MACROS INCLISP LTEMPLATE))  
  (BLOCKS (ALLCALLS ALLCALLS ADDTO MSBINDVAR MSDWIMTRAN MSPRGCALL MSPRGDWIM MSPRGE MSPRGMACRO  
    MSPRGERR MSPRGLAMBDA MSPRGLST MSPRGRECORD MSPRGTEMPLATE MSPRGTEMPLATE1 NLAMBDAFNP  
    (NOLINKFNS . T)  
    (LOCALFREEVARS FNNAME ERRORS FNDEF INCLISP ONLYRELS PARENTCONTEXT TOPVARS PARENT  
      EACHTIME VARS)  
    (GLOBALVARS CLISPARRAY MSERRORFN MSRECORDTRANFLG MSFNDDATA INVISIBLEVARS CLISPARRAY  
      MTEMPLATES USERTEMPLATES MSRECORDTRANFLG NLAMA NLAML DWIMFLG CLISPTRANFLG  
      DWIMESSGAG)  
    (NOLINKFNS . T))  
    (NIL MSINITFNDDATA NLAMBDAFNP MSPRGDWIM (LOCALVARS . T)  
      (GLOBALVARS NLAMA NLAML MSFNDDATA DWIMFLG DWIMESSGAG)))  
  (P (PUTDQ? MSWORDNAME (LAMBDA (X)  
    X)  
  (COMS (VARS (MTEMPLATES (HASHARRAY 160))  
    (USERTEMPLATES (HASHARRAY 10)))  
  (FILEVARS INITIALTEMPLATES)
```

;;; INITIALTEMPLATES is not needed after loading up

```
[P (MAPC INITIALTEMPLATES (FUNCTION (LAMBDA (X)  
  (MAPC (CDR X)  
    (FUNCTION (LAMBDA (Y)  
      (PUTHASH Y (CAR X)  
        MTEMPLATES])  
  (DECLARE%: EVAL@COMPILE DONTCOPY (PROP MACRO LTEMPLATE)))  
(COMS (FNS MSFINDP)  
  (BLOCKS (MSFINDP MSFINDP]))
```

(PUTPROPS MSANALYZE FILETYPE :COMPILE-FILE)

(DEFINEQ

### (VARS

```
[LAMBDA (FN USEDATABASE) (* Imm%: 29-DEC-75 23 22)  
  (CDR (CALLS FN USEDATABASE T]))
```

### (FREEVARS

```
[LAMBDA (FN USEDATABASE) (* Imm%: 5-DEC-75 11 8)  
  (CADDR (CALLS FN USEDATABASE 'FREEVARS]))
```

### (CALLS

```
[LAMBDA (EXPR USEDATABASE VARSFLG) (* rmk%: " 1-AUG-79 09:55")  
  (PROG (FREES FNDEF FLG)  
    [COND
```

```

((AND USEDATABASE (LITATOM EXPR)
  (GETD 'UPDATEFN))
 (UPDATEFN EXPR NIL 'ERROR)
 (RETURN (LIST [AND (NOT VARSFLG)
  (GETRELATION EXPR ' (CALL NOTERROR])
  (AND (NEQ VARSFLG 'FREEVARS)
  (GETRELATION EXPR 'BIND))
  (GETRELATION EXPR ' (USE FREELY])
GETDLP
 (SETQ FNDEF (COND
  [(LITATOM EXPR)
  (OR (GETD (OR (GETP EXPR 'BROKEN)
  EXPR))
  (GETP EXPR 'EXPR)
  (AND (NEQ EXPR (SETQ EXPR (FNCHECK EXPR NIL NIL T)))
  (GO GETDLP])
  (T EXPR)))
 (RETURN (COND
  ((NULL FNDEF)
  NIL)
  ((SUBRP FNDEF)
  NIL)
  ((CCODEP FNDEF)
  (SETQ FNDEF (CALLSCODE FNDEF))
  [OR NOPACKCALLSFLG (for x on (CAR FNDEF) do (FRPLACA X (PACK* ' ; (CAR X)
  ' ;])
  (FRPLACA (CDR FNDEF)
  (NCONC (CADR FNDEF)
  (CAR FNDEF)))
  (SETQ FLG)
  (CALLS3 (CDDR FNDEF))
  (CALLS3 (CDDDR FNDEF))
  (CDR FNDEF))
  [(EXPRP FNDEF)

```

(\* Note that EXPR can be a piece of a function definition, and calls will still work.)

```

(RESETVARS ((MSRECORDTRANFLG T))
  (RETURN (PROG (CALLSDATA LAMFLG)
    [COND
      ((FMEMB (CAR FNDEF)
        LAMBDA SPLST)
        (SETQ LAMFLG T)
        (COND
          ((OR (AND (EQ (CAR (CADDR FNDEF))
            '*))
            (EQ (CADR (CADDR FNDEF))
            'DECLARATIONS%:))
            (EQ (CAR (CADDR FNDEF))
            'CLISP%:))
            (MSPRGDWIM FNDEF EXPR FNDEF)))
          (SELECTQ (CAR FNDEF)
            [(LAMBDA NLAMBDA]
              NIL)
            (SETQ FNDEF (OR (AND COMPILEUSERFN (APPLY* COMPILEUSERFN
              NIL FNDEF))
              FNDEF])
          (SETQ CALLSDATA
            (ALLCALLS
              FNDEF LAMFLG
              [UNION (CONSTANT (MSVBNOTICED 'USE 'FREELY))
                (AND (NEQ VARSFLG 'FREEVARS)
                  (UNION (CONSTANT (MSVBNOTICED 'BIND))
                    (AND (NULL VARSFLG)
                      (CONSTANT (MSVBNOTICED 'CALL
                        'NOTERROR])
                    EXPR T))
                (SETQ FREES (NCONC FREES (COLLECTFNDDATA
                  (CONSTANT (MSVBNOTICED 'USE
                    'FREELY])
                  (RETURN (LIST [COLLECTFNDDATA (CONSTANT (MSVBNOTICED
                    'CALL
                    'NOTERROR])
                    [COLLECTFNDDATA (CONSTANT (MSVBNOTICED
                    'BIND]
                    FREES])
                (T ' ?)])

```

**(COLLECTFNDDATA**

```

[LAMBDA (LST)
  (COND
    ((NLISTP LST)
     (CDR (FASSOC LST CALLSDATA)))
    (T (PROG (VAL)
      (for x in LST do (SETQ VAL (UNION (COLLECTFNDDATA X
        VAL))))

```

(\* Imm "21-DEC-78 22:56")

(RETURN VAL])

(CALLS3

[LAMBDA (LST)

(\* Imm " 6-JUL-78 00:23")
(\* Imm%: 13-DEC-75 4 51)

(PROG (FLG)

[for X on (CAR LST) do (OR (NOT (FMEMB (CAR X)
INVISIBLEVARS))
(SETQ FLG (FRPLACA X]

(COND

(FLG (FRPLACA LST (DREMOVE NIL (CAR LST]))

)

(RPAQQ MSMACROPROPS (ALTMACRO DMACRO BYTEMACRO MACRO))

(RPAQQ NOPACKCALLSFLG NIL)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY

(BLOCK%: CALLS CALLS COLLECTFNDDATA CALLS3 (LOCALFREEVARS CALLSDATA)
(NOLINKFNS . T)

(GLOBALVARS INVISIBLEVARS COMPILEUSERFN NOPACKCALLSFLG LAMBDA SPLST MSRECORDTRANFLG))

(BLOCK%: NIL VARS FREEVARS (LOCALVARS . T))

)

(DECLARE%: EVAL@COMPILE

(RPAQQ MS.VERB.TO.NOTICED

((BIND (NIL BIND ARG))

[CALL (DIRECTLY CALL EFFECT PREDICATE NLAMBDA)

(EFFECT EFFECT)

(INDIRECTLY APPLY STACK)

(NIL CALL ERROR APPLY EFFECT PREDICATE NLAMBDA STACK)

(NOTERROR APPLY CALL EFFECT PREDICATE NLAMBDA STACK)

(PREDICATE PREDICATE)

(TESTING PREDICATE)

(VALUE CALL)

(NLAMBDA NLAMBDA)

(CREATE (NIL CREATE))

(DECLARE (LOCALVARS LOCALVARS)

(NIL LOCALVARS SPECVARS)

(SPECVARS SPECVARS))

(FETCH (NIL FETCH))

(REFERENCE (FIELDS FETCH)

(FREELY REFFREE)

(CL:LOCALLY REF)

(NIL REFFREE REF))

(REPLACE (NIL REPLACE))

(SET (FIELDS FETCH REPLACE)

(FREELY SETFREE)

(CL:LOCALLY SET)

(NIL SETFREE SET))

(SMASH (FIELDS FETCH REPLACE)

(FREELY SMASHFREE)

(CL:LOCALLY SMASH)

(NIL SMASHFREE SMASH))

(TEST (FREELY TESTFREE)

(CL:LOCALLY TEST)

(NIL TESTFREE TEST))

(USE (FIELDS FETCH REPLACE)

(FREELY SETFREE SMASHFREE REFFREE TESTFREE)

(I.S.OPRS CLISP)

(INDIRECTLY LOCALFREEVARS)

(CL:LOCALLY SET SMASH REF TEST)

(NIL SETFREE SET SMASHFREE SMASH REFFREE REF TESTFREE TEST)

(PREDICATE TEST TESTFREE)

(PROPNAME PROP)

(RECORDS RECORD CREATE)

(TESTING TEST TESTFREE)

(VALUE SMASH SMASHFREE REF REFFREE)

(TYPE TYPE)))

(DECLARE%: EVAL@COMPILE

(PUTPROPS MSVBNOTICED MACRO [OPENLAMBDA (VERB MOD)

(CDR (ASSOC MOD (CDR (ASSOC VERB MS.VERB.TO.NOTICED))

)

)

(DEFINEQ

(ALLCALLS

[LAMBDA (FNDEF LAMFLG ONLYRELS FNNAME INTERNALFLG EACHTIME)

; Edited 21-Apr-88 16:31 by jrb:

```
(LET (VARS TOPVARS INCLISP ERRORS (PARENT FNDEF)
      (NOSPELLFLG T))
  (DECLARE (CL:SPECIAL NOSPELLFLG))
  (MSINITFNDA)
  [COND
    (LAMFLG (MSPRGLAMBDA FNDEF 'ARG))
    (T (MSPRGE FNDEF NIL 'RETURN)
      (COND
        (INTERNALFLG MSFNDA)
        (T (for X in MSFNDA when (CDR X) collect (CONS (CAR X)
                                                         (CDR X)]))
```

(MSINITFNDA

[LAMBDA NIL

(\* MSFNDA is an association list of the "noticed" types, e.g. CALL, BIND, etc.
-- the FRPLACD resets the pointer)

(for Y in MSFNDA do (FRPLACD Y NIL])

(MSPRGE

[LAMBDA (EXPR SUPEXPR EVALCONTEXT)

(\* Imm "27-May-86 04:44")

:: analyzes EXPR; SUPEXPR is the parent expression and is used in the SHOWUSE case where we are printing occurrences of various things
:: rather than updating data base; EVALCONTEXT is a type of reference for this expression from the template: SMASH etc

```
(PROG (TEM CALLED CLISP)
  [COND
    ((NLISTP EXPR)
     (RETURN (COND
              ((AND (LITATOM EXPR)
                    EXPR
                    (NEQ EXPR T)
                    (NOT (FMEMB EXPR INVISIBLEVARS))) ; A variable reference
              (COND
                ([OR (FMEMB EXPR VARS)
                    (SOME TOPVARS (FUNCTION (LAMBDA (Z)
                                              ; bound higher up in the function but but used in a functional
                                              ; argument
                                              (COND
                                                ((FMEMB EXPR Z)
                                                 (ADDTO 'LOCALFREEVARS EXPR SUPEXPR)
                                                 T]
                                              ; Things were added to VARS only if they were 'noticeable'
                (SELECTQ EVALCONTEXT
                  ((SMASH TEST SET)
                   (ADDTO EVALCONTEXT EXPR SUPEXPR))
                  (CHANGE (ADDTO 'SET EXPR SUPEXPR))
                  (ADDTO 'REF EXPR SUPEXPR)))
                (T (SELECTQ EVALCONTEXT
                  (SMASH (ADDTO 'SMASHFREE EXPR SUPEXPR))
                  (TEST (ADDTO 'TESTFREE EXPR SUPEXPR))
                  ((SET CHANGE)
                   (ADDTO 'SETFREE EXPR SUPEXPR))
                  (ADDTO 'REFFREE EXPR SUPEXPR]
              (COND
                ((EQ EVALCONTEXT 'SET)
                 (MSPRGERR PARENT))) ; in a 'SET' context, but not a variable
              (COND
                ((LISTP (SETQ CALLED (CAR EXPR)))
                 (MSPRGLAMBDA CALLED NIL (SELECTQ EVALCONTEXT
                  ((TEST EFFECT SMASH)
                   EVALCONTEXT)
                  NIL))
                (SELECTQ (CAR CALLED)
                  (LAMBDA (MSPRGLST (CDR EXPR)
                                EXPR)
                    NIL)
                  (RETURN)))
              [COND
                ((SETQ TEM (LTEMPLATE CALLED))
                 (RETURN (MSPRGTEMPLATE EXPR TEM EVALCONTEXT)
                 (COND
                  ((NOT (FGETD (OR (GETP CALLED 'BROKEN)
                                CALLED)))
                   (COND
                     [[AND DWIMFLG (SETQ TEM (GETPROP CALLED 'CLISPPWD]
                     (SETQ CLISP (MSDWIMTRAN EXPR))
                     (RETURN (COND
                       (CLISP [COND
                         (TEM (SELECTQ (CAR TEM)
                                       (RECORDTRAN (OR (MSPRGRECORD EXPR EVALCONTEXT)
                                                       MSRECORDTRANFLG
                                                       (RETURN))
                                       (* optionally also look at translation)
```



```

((NOT (FMEMB VAR INVISIBLEVARS))
 (ADDO (OR TYPE 'BIND)
  VAR
  (OR EXPR PARENT]
 (SETQ VARS (CONS VAR VARS)))
 (T (MSPRGERR (COND
  ((LITATOM VAR)
   (OR EXPR PARENT))
  (T VAR]))

```

(MSPRGRECORD

[LAMBDA (PRNT CNTXT)

; Edited 8-Apr-88 14:49 by jrb:
(\* ANALYZE RECORD EXPRESSION PRNT -
RETURN NIL IF ANALYZED SUCCESSFULLY)

```

(PROG (Z)
 (MSPRGTEMPLATE
  PRNT
  (SELECTQ (CAR PRNT)
   ((create CREATE)
    (ADDO 'CREATE (CADR PRNT)
     PRNT)
    (SETQ Z (CDDR PRNT))
    [while Z do (COND
     ([EQ 'RECORDTRAN (CAR (GETPROP (CAR Z)
      'CLISPPWORD]
      (* e.g. USING or COPYING)

      (MSPRGE (CADR Z)
       PRNT
       (SELECTQ (CAR Z)
        ((smashing SMASHING)
         'SMASH)
         NIL))
       (SETQ Z (CDDR Z)))
     (EQ (CADR Z)
      '_)
      (* If dwimified correctly, the fields should be separated by _'s)
      (ADDO 'REPLACE (CAR Z)
       PRNT)
      (MSPRGE (CADDR Z)
       PRNT)
       (SETQ Z (CDDDR Z)))
     (EQ (CAAR Z)
      'SETQ)
      (* partially dwimified)
      (ADDO 'REPLACE (CADAR Z)
       PRNT)
      (MSPRGE (CADDAR Z)
       PRNT)
       (SETQ Z (CDR Z)))
     (T
      (* shouldn't happen, but)
      (MSPRGE (CAR Z)
       PRNT)
      (SETQ Z (CDR Z]
      (RETURN))
      ((fetch FETCH ffetch FFETCH)
       [LET [(OF? (OR (EQ (CL:THIRD PRNT)
        'OF)
        (EQ (CL:THIRD PRNT)
        'of]
        (COND
         [(EQ CNTXT 'CHANGE)
          `(NIL (IF LISTP (BOTH (NIL |..| FETCH (BOTH FETCH REPLACE))
           (|..| RECORD NIL))
            (BOTH FETCH REPLACE))
           ,@(if OF?
            then '(NIL EVAL |..| PPE)
            else '(EVAL |..| PPE)]
          (T `(NIL (IF LISTP (BOTH (NIL |..| FETCH)
           (|..| RECORD NIL))
            FETCH)
           ,@(if OF?
            then '(NIL EVAL |..| PPE)
            else '(EVAL |..| PPE)]
          ((REPLACE /REPLACE replace /replace freplace FREPLACE)
           (LET* [(OF? (OR (EQ (CL:THIRD PRNT)
            'OF)
            (EQ (CL:THIRD PRNT)
            'of]
            (WITHSLOT (if OF?
             then (CL:FIFTH PRNT)
             else (CL:FOURTH PRNT)))
            (WITH? (OR (EQ WITHSLOT 'WITH)
             (EQ WITHSLOT 'with]
            `(NIL (IF LISTP (BOTH (NIL |..| FETCH REPLACE)
             (|..| RECORD NIL))
              REPLACE)
              ,@(if OF?
               then '(NIL)

```



```

      T)
      NIL))
    (EQ X T)
    (NULL X)
    (PROGN
      (ADDTO 'ERROR 'apply PARENT)
      (MSPRGE X PARENT 'FUNCTIONAL])
      (* arbitrary expression as functional argument)
(STACK
  (OR (AND (LISTP X)
    (SELECTQ (CAR X)
      ((FUNCTION QUOTE)
        (ADDTO 'STACK (CADR X)
          PARENT)
        T)
      NIL))
    (PROGN (ADDTO 'ERROR 'stackfn PARENT)
      (MSPRGE X PARENT 'EVAL]))
      (* arg to stack fn, e.g. RETFROM)
(PROP (COND
  ((AND (LISTP X)
    (EQ (CAR X)
      'QUOTE))
    (for Y inside (CADR X) do (ADDTO 'PROP Y PARENT)))
  (T (MSPRGE X PARENT TEMPLATE))))
(NIL
  NIL)
(RETURN
  (MSPRGTEMPLATE1 X (SELECTQ PARENTCONTEXT
    ((TEST EFFECT)
      PARENTCONTEXT)
    'EVAL)))
      (* not used)
      (* this is sometimes the value of PARENT expression)
(TESTRETURN
  (MSPRGTEMPLATE1 X (SELECTQ PARENTCONTEXT
    (TEST PARENTCONTEXT)
    'EVAL)))
      (* if PARENT is tested, then so is this)
(BIND (MSBINDVAR X))
(LAMBDA (MSPRGLAMBDA X))
(PPE
  (COND
    (X (MSPRGERR PARENT)
      (MSPRGLST X PARENT))))
      (* paren error if not NIL)
(CALL (MSPRGCALL X PARENT PARENTCONTEXT))
(EVALQT [COND
  ((EQ (CAR X)
    'QUOTE)
    (MSPRGTEMPLATE (CADR X)
      ' (REMOTE EVAL)
      PARENT))
  (T (MSPRGE X PARENT 'EVAL]))
  (ADDTO TEMPLATE X PARENT T)))
(T (SELECTQ (CAR TEMPLATE)
  (IF [PROG ((EXPR X))
    (DECLARE (SPECVARS EXPR))
    (MSPRGTEMPLATE1 X (COND
      ((COND
        ((LISTP (CADR TEMPLATE))
          (* ASSERT%: ((REMOTE EVAL) EXPR))
          (EVAL (CADR TEMPLATE)))
        (T (APPLY* (CADR TEMPLATE)
          X)))
          (CADDR TEMPLATE))
        (T (CADDRR TEMPLATE))
      ))
    (|. . | [COND
  [(AND (CADR TEMPLATE)
    (NULL (CDDR TEMPLATE)))]
      (* Special case to handle most common cases)
    (MAPC X (FUNCTION (LAMBDA (X)
      (MSPRGTEMPLATE1 X (CADR TEMPLATE)
        (T (FRPTQ (IDIFFERENCE (LENGTH X)
          (LENGTH (CDDR TEMPLATE))))
          (MSPRGTEMPLATE1 (CAR X)
            (CADR TEMPLATE))
          (SETQ X (CDR X)))
          (MSPRGTEMPLATE1 X (CDDR TEMPLATE))
        (MACRO (ADDTO 'CALL (CAR X)
          PARENT)
          (MSPRGMACRO X (CDR TEMPLATE)))
        (BOTH (MSPRGTEMPLATE1 X (CADR TEMPLATE))
          (MSPRGTEMPLATE1 X (CADDR TEMPLATE)))
        (@ [PROG ((EXPR X))
          (DECLARE (SPECVARS EXPR))
          (MSPRGTEMPLATE1 (EVAL (CADR TEMPLATE))
            (EVAL (CADDR TEMPLATE))
            (REMOTE (PROG (VARS (TOPVARS (CONS VARS TOPVARS))))
              (MSPRGTEMPLATE1 X (CADR TEMPLATE))))
          (KEYWORDS
            ;; KEYWORDS list of keys...
            ;; Specifies list of legal keywords
            ;; (FOO (LAMBDA... (BAR :BAZ DREK))) is recorded in the database as

```



;; (ADDTO 'KEYCALLS '(BAR . :BAZ) '(BAR :BAZ DREK))

;; i.e. there is a table for each keyword relating functions that call functions specifying them.

```
[LET [(LEGAL-KEYS (OR (CDR TEMPLATE)
                     (GETRELATION (CAR PARENT)
                                   'KEYACCEPT]
      (while X bind (ALLOW-OTHER-KEYS _ (MEMQ '&ALLOW-OTHER-KEYS LEGAL-KEYS))
                    KEYSUSED?
      do (if (AND (CL:KEYWORDP (CAR X))
                 (OR ALLOW-OTHER-KEYS (MEMQ (CAR X)
                                             LEGAL-KEYS)))
          then (ADDTO 'KEYSPECIFY (CAR X)
                     PARENT)
          (SETQ KEYSUSED? T)
          else (MSPRGERR PARENT)) ; log bogus keyword as ppe
      (pop X)
      (if X
          then (MSPRGTEMPLATE1 (CAR X)
                               'EVAL)
          (pop X)
          else (MSPRGERR PARENT)) ; log no value for keyword as ppe
      )
      finally (if KEYSUSED?
                 then (ADDTO 'KEYCALL (CAR PARENT)
                             PARENT]))
(COND
  ((LISTP X)
   (MSPRGTEMPLATE1 (CAR X)
                   (CAR TEMPLATE))
   (MSPRGTEMPLATE1 (CDR X)
                   (CDR TEMPLATE]))
```

**(MSPRGTEMPLATE**

```
[LAMBDA (PARENT TEMPLATE PARENTCONTEXT)
  (BLOCK)
```

(\* Imm "23-Jul-86 00:15")

(\* "Masterscope should block every once and a while. This is one place to do it.")

```
(PROG ((VARS VARS)
      TEM)
  (COND
    [(EQ TEMPLATE 'MACRO)
     (COND
       ((SETQ TEM (GETMACROPROP (CAR PARENT)
                                MSMACROPROPS))
        (MSPRGMACRO PARENT TEM))
       (T (MSPRGTEMPLATE1 PARENT '(CALL |..| EVAL)
                           (T (MSPRGTEMPLATE1 PARENT TEMPLATE))
```

**(MSPRGLAMBDA**

```
[LAMBDA (EXPR FLG TYPE)
  (SELECTQ (CAR (LISTP EXPR))
    (CL:LAMBDA [LET
```

; Edited 3-Jun-88 10:23 by jrb:

```
  ((PARENT EXPR)
   (PARENTCONTEXT TYPE)
   (VARS VARS))
  [bind (ARGS _ (CADR EXPR))
    ARG
    (EVALCOUNT _ 0)
    KEYS? KEYLIST while ARGS
  do (SETQ ARG (pop ARGS))
    ;; We can be in one of two states - keyword scanning or not
    [COND
      [(CL:SYMBOLP ARG)
       ;; Check and see if it's a keyword thingy
       (COND
         ((EQ ARG '&KEY)
          (SETQ KEYS? T))
         [(FMEMB ARG CL:LAMBDA-LIST-KEYWORDS)
          (SETQ KEYS? NIL)
          (if (EQ ARG '&ALLOW-OTHER-KEYS)
              then (ADDTO 'KEYACCEPT (CAR (push KEYLIST '&ALLOW-OTHER-KEYS))
                          (CADR EXPR))
              (T (if KEYS?
                    then (ADDTO 'KEYACCEPT (CAR (push KEYLIST (MAKE-KEYWORD ARG)))
                                (CADR EXPR))
                    else (CL:INCF EVALCOUNT))
              (MSBINDVAR ARG]
      [(CL:CONSP ARG)
       (MSPRGTEMPLATE1 (CADR ARG)
                       'EVAL)
       (if KEYS?
           then (if (CL:SYMBOLP (CAR ARG))
                   then (MSBINDVAR (CAR ARG))
```

; Strangely enough they all EVAL their CADRs.

```

        (ADDDO 'KEYACCEPT [CAR (push KEYLIST (MAKE-KEYWORD (CAR ARG)
            (CADR EXPR))
        else (ADDDO 'KEYACCEPT [CAR (push KEYLIST (MAKE-KEYWORD (CAAR ARG)
            (CADR EXPR))
        (MSBINDVAR (CADAR ARG)))
    else (CL:IF (CL:SYMBOLP (CAR ARG))
        (MSBINDVAR (CAR ARG))
        (MSBINDVAR (CADAR ARG)))
        (OR (NULL (CDDR ARG))
            (NOT (CL:SYMBOLP (CADDR ARG)))
            (MSBINDVAR ARG])
    (T (MSPRGTEMPLATE1 ARG 'PPE]
finally (if KEYLIST
    then
        ;; Look at old template; if there isn't one or it looks like one we put out
        ;; (of the form (EVAL* KEYWORDS keys...))
        ;; replace it with a new template (and somehow mark everything that calls FNAME for reanalysis
        [LET ((OLDTEMP (GETTEMPLATE FNAME))
            (EC EVALCOUNT))
            ;; First pop off all the EVALS at the front and count them
            (while (EQ (CAR OLDTEMP)
                'EVAL)
                do (CL:DECF EC)
                    (pop OLDTEMP))
            (if (OR (NULL OLDTEMP)
                (EQ (CAR OLDTEMP)
                    'KEYWORDS))
                then (pop OLDTEMP)
                    (if (AND (CL:ZEROP EC)
                        (NULL (CL:SET-DIFFERENCE OLDTEMP KEYLIST))
                        (NULL (CL:SET-DIFFERENCE KEYLIST OLDTEMP)))
                        then
                            ; it matches, don't replace it
                            NIL
                        else
                            ; It looks like one of ours; replace it
                            (SETQ KEYLIST (CONS 'KEYWORDS (CL:NREVERSE KEYLIST)))
                            (while (CL:PLUSP EVALCOUNT)
                                do (push KEYLIST 'EVAL)
                                    (CL:DECF EVALCOUNT))
                            (SETTEMPLATE FNAME KEYLIST)
                            ;; These templates shouldn't show up in FILES?, since they're solely for
                            ;; Masterscope internal use...
                            (UNMARKASCHANGED FNAME 'TEMPLATES)
                            ;; The SETTEMPLATE call marked everyone who calls FNAME to be
                            ;; reanalyzed; it also marked FNAME - this gets rid of that
                            (PUTHASH FNAME NIL MSCHANGEDARRAY]
                    else
                        ;; It's possible that we created an old template for this function and it no longer has keywords, so we
                        ;; MAY need to delete it
                        (LET ((OLDTEMP (GETTEMPLATE FNAME)))
                            (while (EQ (CAR OLDTEMP)
                                'EVAL)
                                do (pop OLDTEMP))
                            (if (EQ (CAR OLDTEMP)
                                'KEYWORDS)
                                then (SETTEMPLATE FNAME NIL)
                                    ;; These templates shouldn't show up in FILES?, since they're solely for Masterscope
                                    ;; internal use...
                                    (UNMARKASCHANGED FNAME 'TEMPLATES)
                                    ;; The SETTEMPLATE call marked everyone who calls FNAME to be reanalyzed; it
                                    ;; also marked FNAME - this gets rid of that
                                    (PUTHASH FNAME NIL MSCHANGEDARRAY]
                (MSPRGTEMPLATE1 (CDDR EXPR)
                    '(|..| EVAL RETURN])
        ([LAMBDA NLAMBDA OPENNLAMBDA]
            (MSPRGTEMPLATE EXPR '(NIL (IF LISTP (|..| BIND)
                (IF (PROGN EXPR) BIND))
                    |..| EFFECT RETURN)
                TYPE))
        (PROG (CLISP TEM)
            (COND
                ((AND (SETQ TEM (ASSOC (CAR EXPR)
                    LAMBDATRANFNS))
                    (SETQ CLISP (CL:FUNCALL (CADR TEM)
                        EXPR)))
                    (PROG ((INCLISP (INCLISP EXPR)))
                        (MSPRGLAMBDA CLISP FLG T)))
                ((AND DWIMFLG (SETQ CLISP (MSDWIMTRAN EXPR))) (* has a CLISP translation (e.g. DLAMBDA))
                    (PROG ((INCLISP (INCLISP EXPR)))
                        (MSPRGLAMBDA CLISP FLG TYPE)))
                (T (MSPRGERR EXPR)
                    (MSPRGE EXPR])

```

**(MSPRGLST**

```
[LAMBDA (L PARNT CNTX)
  (for X in L do (MSPRGE X PARNT CNTX])
```

(\* Imm "27-JUN-78 01:57")

**(ADDTO**

```
[LAMBDA (RELATION WHAT PRNT FLG)
  (PROG ((PTR (FASSOC RELATION MSFNDDATA)))
    [OR PTR (COND
      (FLG (RETURN))
      (T (SHOULDNT])
    (OR (NULL ONLYRELS)
      (FMEMB RELATION ONLYRELS)
      (RETURN))
    (AND EACHTIME (EQMEMB RELATION (CAR EACHTIME))
      (APPLY* (CADR EACHTIME)
        WHAT
        (CADDR EACHTIME)
        (CADDRR EACHTIME)
        PRNT INCLISP))
    LP (COND
      ((NULL (CDR PTR))
        (FRPLACD PTR (LIST WHAT)))
      ((EQ (CAR (SETQ PTR (CDR PTR)))
        WHAT)
        (RETURN))
      (T (GO LP]))
```

(\* Imm "24-DEC-78 11:51")

**(NLAMBDADFNP**

```
[LAMBDA (FN)
  (AND [NOT (EQMEMB 'EVAL (GETPROP FN 'INFO)
    (COND
      ((OR (FGETD (SETQ FN (OR (GETPROP FN 'BROKEN)
        FN)))
        (SETQ FN (GETLIS FN ' (EXPR CODE]
```

(\* Imm "26-Mar-85 12:37")

(\* if the function is defined, check its argtype to tell you whether it is NLAMBDA or LAMBDA)

```
(SELECTQ (ARGTYPE FN)
  ((1 3)
   T)
  NIL))
(T
  (OR (FMEMB FN NLAMA)
    (FMEMB FN NLAML]))
```

(\* otherwise, rely on NLAMA or NLAML)

**(MSPRGDWIM**

```
[LAMBDA (X FN DEF)
  (AND DWIMFLG (LET ((NOSPPELLFLG T)
    FILEPKGFLG)
    (DECLARE (SPECVARS NOSPPELLFLG))
    (RESETVARS ((DWIMESSGAG T))
```

; Edited 8-Apr-88 11:55 by jrb:

;; JRB Made these RESETVARS to placate the PavCompiler

```
(PROG (LISPXHIST) (* ASSERT%: ((REMOTE EVAL) DWIMESSGAG FILEPKGFLG
  NOSPPELLFLG)
  (DWIMIFYO X (OR (AND (LITATOM FN)
    FN)
    '?))
  VARS DEF])
```

**(MSDWIMTRAN**

```
[LAMBDA (EXPR)
  (AND DWIMFLG (COND
    ((AND CLISPARRAY (GETHASH EXPR CLISPARRAY)))
    ((AND CLISPTRANFLG (EQ (CAR (LISTP EXPR))
      CLISPTRANFLG))
      (CADR EXPR))
    (T (MSPRGDWIM EXPR FNAME FNDEF)
      (OR (AND CLISPARRAY (GETHASH EXPR CLISPARRAY))
        (AND CLISPTRANFLG (EQ (CAR (LISTP EXPR))
          CLISPTRANFLG))
        (CADR EXPR]))
```

(\* DD%: "28-DEC-81 13:46")

**(RPAQQ MSFNDDATA**

```
((BIND)
(CAL)
(EFFECT)
(PREDICATE)
(CLISP)
```

```

(PROP)
(SETFREE)
(SET)
(SMASHFREE)
(SMASH)
(REFFREE)
(REF)
(FETCH)
(REPLACE)
(RECORD)
(ERROR)
(ARG)
(CREATE)
(LOCALVARS)
(SPECVARS)
(APPLY)
(TESTFREE)
(TEST)
(LOCALFREEVARS)
[NLAMBDA]
(TYPE)
(STACK)
(KEYACCEPT)
(KEYSPECIFY)
(KEYCALL)
(FLET)
(LABEL)
(MACROLET)
(COMPILER-LET))

```

(RPAQQ MSERRORFN ppe)

(RPAQQ MSRECORDTRANFLG NIL)

(ADDOVAR INVISIBLEVARS \$\$1 \$\$2 \$\$3 \$\$4 \$\$5 \$\$6 \$\$7 \$\$8 \$\$9 \$\$10 \$\$11 \$\$12 \$\$13 \$\$14 \$\$15 \$\$16 \$\$17)

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE

```

(PUTPROPS INCLISP MACRO ((.X.)
  (COND
    ((AND INCLISP EACHTIME (NOT (MSFINDP INCLISP .X.)))
      INCLISP)
    (T .X.)))

```

```

(PUTPROPS LTEMPLATE MACRO [LAMBDA (Y)
  (DECLARE (LOCALVARS Y)
    (AND [NEQ T (SETQ Y (OR (GETHASH Y USERTEMPLATES)
      (GETHASH Y MSTEMPLATES)
    Y])

```

)  
)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY

```

(BLOCK%: ALLCALLS ALLCALLS ADDTO MSBINDVAR MSDWIMTRAN MSPRCALL MSPRGDWIM MSPRGE MSPRGMACRO MSPRGERR MSPRGLAMBDA
  MSPRGLST MSPRGRECORD MSPRGTEMPLATE MSPRGTEMPLATE1 NLAMBDAFNP (NOLINKFNS . T)
  (LOCALFREEVARS FNNAME ERRORS FNDEF INCLISP ONLYRELS PARENTCONTEXT TOPVARS PARENT EACHTIME VARS)
  (GLOBALVARS CLISPARRAY MSERRORFN MSRECORDTRANFLG MSFNDATA INVISIBLEVARS CLISPARRAY MSTEMPLATES
    USERTEMPLATES MSRECORDTRANFLG NLAMA NLAML DWIMFLG CLISPTRANFLG DWIMESSGAG)
  (NOLINKFNS . T))

```

```

(BLOCK%: NIL MSINITFNDATA NLAMBDAFNP MSPRGDWIM (LOCALVARS . T)
  (GLOBALVARS NLAMA NLAML MSFNDATA DWIMFLG DWIMESSGAG))
)

```

```

[PUTDQ? MSWORDNAME (LAMBDA (X)
  X]

```

(RPAQ MSTEMPLATES (HASHARRAY 160))

(RPAQ USERTEMPLATES (HASHARRAY 10))

```

(RPAQQ INITIALTEMPLATES
  (((CALL (REMOTE (IF LITATOM CALL LAMBDA))
    (IF LITATOM EVAL NIL))
    FUNCTION)
  ((NIL NIL |..| EVAL RETURN)
  CL:BLOCK)
  ((CALL |..| EVAL)
  CL:CATCH CL:THROW)
  ((NIL NIL EVAL)
  CL:RETURN-FROM)
  ([IF (EQ (CADR EXPR)
    'ASSERT%:)
    (NIL NIL |..| (IF LISTP (@ (CDR EXPR)

```

```

                                (LIST '|..| (MSWORDNAME (CAR EXPR]
*)
(NIL (BOTH (|..| (IF LISTP (NIL EVAL |..| EFFECT)
              NIL))
        (|..| (IF LISTP (BIND)
              BIND)))
      |..| EFFECT RETURN)
LET CL:COMPILER-LET)
(NIL (|..| (IF LISTP (BIND EVAL . PPE)
              BIND))
      |..| EFFECT RETURN)
LET*)
(NIL |..| (IF LISTP EFFECT))
CL:TAGBODY)
(NIL (BOTH (|..| (IF LISTP (NIL EVAL |..| EFFECT)
              NIL))
        (|..| (IF LISTP (BIND)
              BIND)))
      |..|
      (IF LISTP EFFECT))
PROG)
(MACRO RESETVARS)
((CALL EVAL)
 XNLSETQ NLSETQ ERSETQ)
((CALL |..| EVAL)
 RESETFORM FRPTQ)
((CALL EVAL EVAL FUNCTIONAL FUNCTIONAL . PPE)
 MAP2C)
((CALL EVAL EVAL SMASH . PPE)
 /DSUBST DSUBST)
((CALL EVAL FUNCTION FUNCTION . PPE)
 MAPCAR MAPCON MAPCONC MAPLIST SUBSET EVERY NOTEVERY ANY NOTANY SOME MAPC MAP)
((CALL EVAL FUNCTIONAL . PPE)
 MAPHASH)
((CALL EVAL PROP . PPE)
 GETP GETLIS GET GETPROP LISTGET LISTGET1 REMPROP /REMPROP)
((CALL EVAL PROP EVAL . PPE)
 PUT /PUT PUTPROP /PUTPROP LISTPUT LISTPUT1)
((CALL EVAL SMASH . PPE)
 /ATTACH ATTACH)
((CALL FUNCTIONAL . PPE)
 MAPATOMS)
((CALL FUNCTIONAL |..| EVAL)
 APPLY* BLKAPPLY* APPLY BLKAPPLY)
((CALL EVAL SMASH . PPE)
 DREMOVE /DREMOVE)
((CALL SET EVAL EVAL . PPE)
 RESETVAR)
((CALL SET EVAL . PPE)
 SETN)
((CALL SMASH . PPE)
 DREVERSE)
((CALL SMASH EVAL . PPE)
 RPLACD /RPLACD RPLACA /RPLACA RPLNODE2 /RPLNODE2 FRPLACD FRPLNODE2 TCONC /TCONC LCONC /LCONC NCONC1
 /NCONC1 FRPLACA)
((CALL SMASH EVAL EVAL . PPE)
 RPLNODE FRPLNODE /RPLNODE)
((CALL SMASH FUNCTIONAL . PPE)
 SORT)
((CALL (BOTH SET EVAL) . PPE)
 ADDIVAR SUB1VAR)
((CALL (IF NULL NIL (IF ATOM SET EVAL))
      EVAL . PPE)
 RESETSAVE)
((CALL (IF (EQ (CAR (LISTP EXPR))
              'QUOTE)
          (NIL SET)
          EVAL)
      EVAL . PPE)
 SET /SET SETTOPVAL /SETTOPVAL SETATOMVAL /SETATOMVAL)
((CALL (IF (EQ (CAR (LISTP EXPR))
              'QUOTE)
          (NIL SET)
          EVAL)
      EVAL EVAL EVAL . PPE)
 SAVESET)
((CALL (IF (EQ (CAR (LISTP EXPR))
              'QUOTE)
          (NIL EVAL)
          EVAL)
      |..| EVAL)
 GETATOMVAL EVAL EVALV)
(NIL |..| TESTRETURN RETURN)
OR)
(NIL |..| TEST RETURN)
AND)
(NIL |..| EFFECT RETURN)

```

```

PROGN
((NIL TEST RETURN RETURN)
CL:IF)
((NIL |..| (IF CDR (TEST |..| EFFECT RETURN)
              (TESTRETURN . PPE)))
COND)
([CALL |..| (@ EXPR (CONS NIL (SELECTQ (CAR (LISTP EXPR))
                                       (LOCALVARS '(IF LISTP (|..| LOCALVARS)
                                                    LOCALVARS))
                                       (SPECVARS '(IF LISTP (|..| SPECVARS)
                                                    SPECVARS))
                                       NIL])
DECLARE)
((NIL RETURN)
CLISP% )
((NIL EVAL . PPE)
LISTP NLISTP RETURN)
((NIL TEST . PPE)
NOT NULL)
((CALL EVAL |..| (NIL |..| EFFECT RETURN)
RETURN)
SELECTQ SELCHARQ)
((CALL EVAL |..| (EVAL |..| EFFECT RETURN)
RETURN)
SELECTC)
((CALL EVAL |..| ((IF LISTP (|..| EVAL)
                          EVAL)
                  |..| EFFECT RETURN)
RETURN)
SELECT)
((NIL EVAL EVAL . PPE)
EQ NEQ)
((NIL NIL . PPE)
QUOTE GO)
((NIL EVAL . PPE)
CAR CDR CAAR CADR CDAR CDDR CAAAR CAADR CADAR CDAAR CADDR CDADR CDDAR CDDDR CAAAAR CAAADR CAADAR CAADDR
CADAAR CADADR CADDAR CADDR CDAAR CDAADR CDADAR CDADDR CDDAAR CDDADR CDDDR CDDDDR)
((NIL RETURN |..| EFFECT)
PROG1)
((NIL SET NIL . PPE)
SETQQ)
((NIL SET EVAL . PPE)
SETQ ADV-SETQ SAVESETQ)
([@ 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])
CL:SETQ CL:SETF)
((CALL EVAL (BOTH (@ 'RPTN 'BIND)
                  RETURN) . PPE)
RPTQ)
((CALL EVALQT |..| EVAL)
EVAL ERRORSET)
((BOTH [IF (EQ (CAR (LISTP (CADDR EXPR)))
              'QUOTE)
        (NIL NIL (NIL (|..| (BIND]
                          (CALL EVALQT EVAL . PPE))
EVALA)
((CALL EVALQT STACK STACK EVAL EVAL . PPE)
ENVEVAL)
((CALL FUNCTIONAL EVALQT STACK STACK EVAL EVAL . PPE)
ENVAPPLY)
((CALL STACK EVAL EVAL EVAL . PPE)
STKAPPLY)
((CALL STACK EVALQT EVAL EVAL . PPE)
RETEVAL STKEVAL)
((CALL STACK EVAL EVAL . PPE)
RETFROM RETTO)
((NIL NIL RETURN)
THE))

```

;;; INITIALTEMPLATES is not needed after loading up

```

[MAPC INITIALTEMPLATES (FUNCTION (LAMBDA (X)
                                (MAPC (CDR X)
                                      (FUNCTION (LAMBDA (Y)
                                                (PUTHASH Y (CAR X)
                                                         MTEMPLATES])

```

(DECLARE%: EVAL@COMPILE DONTCOPY

```

{MEDLEY}<loops>obsolete>MSANALYZE.;1

(PUTPROPS LTEMPLATE MACRO [LAMBDA (Y)
      (DECLARE (LOCALVARS Y))
      (AND [NEQ T (SETQ Y (OR (GETHASH Y USERTEMPLATES)
                              (GETHASH Y MSTEMPLATES)
                              Y]))
      )
)

(DEFINEQ
(MSFINDP
  [LAMBDA (STRUC SUB)
    (PROG NIL
      LP (RETURN (OR (EQ SUB STRUC)
                    (AND (LISTP STRUC)
                        (OR (MSFINDP (CAR STRUC)
                                      SUB)
                            (PROGN (SETQ STRUC (CDR STRUC))
                                    (GO LP]))
                    )
      )
)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY

(BLOCK%: MSFINDP MSFINDP)
)

(PUTPROPS MSANALYZE COPYRIGHT ("Xerox Corporation" T 1982 1983 1984 1985 1986 1988))

```

(\* Imm "14-Aug-84 16:38")

---

**FUNCTION INDEX**

ADDTO .....	11	FREEVARS .....	1	MSPRCALL .....	5	MSPRGLST .....	11	NLAMBDANFP .....	11
ALLCALLS .....	3	MSBINDVAR .....	5	MSPRGDWIM .....	11	MSPRGMACRO .....	5	VAR .....	1
CALLS .....	1	MSDWIMTRAN .....	11	MSPRGE .....	4	MSPRGRECORD .....	6		
CALLS3 .....	3	MSFINDP .....	15	MSPRGERR .....	7	MSPRGTEMPLATE .....	9		
COLLECTFNDA .....	2	MSINITFNDA .....	4	MSPRGLAMBDA .....	9	MSPRGTEMPLATE1 .....	7		

---

**VARIABLE INDEX**

INITIALTEMPLATES .....	12	MSERRORFN .....	12	MSRECORDTRANFLG .....	12	USERTEMPLATES .....	12
INVISIBLEVARS .....	12	MSFNDA .....	11	MSTEMPLATES .....	12		
MS.VERB.TO.NOTICED .....	3	MSMACROPROPS .....	3	NOPACKCALLSFLG .....	3		

---

**MACRO INDEX**

INCLISP .....	12	LTEMPLATE .....	12,15	MSVBNOTICED .....	3
---------------	----	-----------------	-------	-------------------	---

---

**PROPERTY INDEX**

MSANALYZE .....	1
-----------------	---

---