```
(RPAQQ EDITINTERFACECOMS
        [;; This is John Wozencraft's simplified edit interface, created December 1990.
        (PROP (Definition-for-EDITL Definition-for-EDITE Definition-for-EDITDATE)
              TELETYPE DISPLAY)
        (GLOBALVARS *LAST-DF* *LAST-DV* *LAST-DC* *LAST-DP*)
        (INITVARS (*LAST-DF*)
                (*LAST-DV*)
                (*LAST-DC*)
                (*LAST-DP*))
        (INITVARS (*EDITMODE* 'TELETYPE)
                (*DISPLAY-EDITOR*))
        ;; init *EDITMODE* to TELETYPE, since that's the only editor we know is loaded.  other editors should set *DISPLAY-EDITOR* and call
        ;; EDITMODE as appropriate.
        (VARS DUMMY-EDIT-FUNCTION-BODY)
        (VARIABLES *ED-OFFERS-PROPERTY-LIST* XCL::ED-LAST-INFO)
        (FUNCTIONS ED INSTALL-PROTOTYPE-DEFN)
        (FNS EDITDEF.FNS EDITF EDITFB EDITFNS EDITLOADFNS? EDITMODE EDITP EDITV DC DF DP DV EDITPROP EF EP EV
              EDITE EDITL)
        (COMS  ;; Time stamp on functions when edited

                ;; User enables this by an (ADDTOVAR INITIALSLIST (USERNAME )) in his INIT.LISP.  E.g.  (ADDTOVAR INITIALSLIST
                ;; (MASINTER )) The date fixup is enabled by the variable INITIALS.  The function SETINITIALS sets INITIALS from INITIALSLIST
                ;; and USERNAME at load time, and after a sysin.

                (FNS NEW/EDITDATE FIXEDITDATE EDITDATE? EDITDATE SETINITIALS)
                (INITVARS (INITIALS)
                        (INITIALSLST)
                        (DEFAULTINITIALS T))
                (INITVARS (*REPLACE-OLD-EDIT-DATES* NIL))
                (P (MOVD? 'EDITDATE 'TTY/EDITDATE))
                (COMS                                            ; Moved from FILEPKG
                        [VARS (EDITDATE-ARGLIST-DEFINERS '(FUNCTIONS TYPES))
                                (EDITDATE-NAME-DEFINERS '(STRUCTURES VARIABLES]
                        (GLOBALVARS EDITDATE-ARGLIST-DEFINERS EDITDATE-NAME-DEFINERS)))
        [INITVARS (COMMON-SOURCE-MANAGER-TYPES '(FUNCTIONS VARIABLES STRUCTURES TYPES SETFS OPTIMIZERS]
        (PROP FILETYPE EDITINTERFACE)
        (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS
                (ADDVARS (NLAMA EV EP EF DV DP DF DC EDITV EDITP EDITFNS EDITF)
                        (NLAML)
                        (LAMA])
```

;; This is John Wozencraft's simplified edit interface, created December 1990.

```
(PUTPROPS TELETYPE Definition-for-EDITL TTY/EDITL)

(PUTPROPS DISPLAY Definition-for-EDITL TTY/EDITL)

(PUTPROPS TELETYPE Definition-for-EDITE TTY/EDITE)

(PUTPROPS DISPLAY Definition-for-EDITE TTY/EDITE)

(PUTPROPS TELETYPE Definition-for-EDITDATE TTY/EDITDATE)

(PUTPROPS DISPLAY Definition-for-EDITDATE TTY/EDITDATE)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS *LAST-DF* *LAST-DV* *LAST-DC* *LAST-DP*)
)

(RPAQ? *LAST-DF* )

(RPAQ? *LAST-DV* )

(RPAQ? *LAST-DC* )

(RPAQ? *LAST-DP* )
```

```
(RPAQ? *EDITMODE* 'TELETYPE)

(RPAQ? *DISPLAY-EDITOR* )
```

;; init *EDITMODE* to TELETYPE, since that's the only editor we know is loaded. other editors should set *DISPLAY-EDITOR* and call EDITMODE as
;; appropriate.

```
(RPAQQ DUMMY-EDIT-FUNCTION-BODY [LAMBDA (ARGS |...|)
                                              BODY])


(CL:DEFVAR *ED-OFFERS-PROPERTY-LIST* T
    "Controls whether ED offers property list as an editable aspect")


(DEFGLOBALVAR XCL::ED-LAST-INFO NIL
    "used in ED to stash last call info so (ED NIL) will restart last edit")


(CL:DEFUN ED (CL::NAME CL::OPTIONS)                                      ; Edited  5-Jul-88 16:03 by woz
```

;;; Standard Common Lisp editor entry. CLtL say's ED does something reasonable when passed a pathname. We coerce name into something that
;;; might be the name of something with an IL:FILES definition, & try to edit that. Then save call info in ED-LAST-INFO, so (ED) will start last edit over
;;; again.

```
    (CL:UNLESS (CL:LISTP CL::OPTIONS)
        (CL:SETQ CL::OPTIONS (LIST CL::OPTIONS)))
    (CL:WHEN (CL:PATHNAMEP CL::NAME)
        (CL:SETQ CL::NAME (NAMEFIELD (CL:STRING-UPCASE (CL:NAMESTRING CL::NAME))
                                   T))
        (CL:PUSHNEW 'FILES CL::OPTIONS))
    [COND
        (CL::NAME (CL:SETQ XCL::ED-LAST-INFO (CONS CL::NAME CL::OPTIONS)))
        (T (CL:WHEN (NULL XCL::ED-LAST-INFO)
               (CL:FORMAT T "Sorry, there is no previous edit to restart.")
               (CL:RETURN-FROM ED NIL))
          (CL:SETQ CL::NAME (CAR XCL::ED-LAST-INFO))
          (CL:SETQ CL::OPTIONS (CL:APPEND (CDR XCL::ED-LAST-INFO)
                                      CL::OPTIONS]
    (LET* ((CL::FROM-DISPLAY (OR (EQ CL::OPTIONS T)
                                 (CL:MEMBER :DISPLAY CL::OPTIONS)
                                 (CL:MEMBER 'DISPLAY CL::OPTIONS)))
           (CL::GIVEN-TYPES (for X inside CL::OPTIONS when (NEQ X T) bind TYPE
                                 when (CL:SETQ TYPE (GETFILEPKGTYPE X 'TYPES T CL::NAME)) collect TYPE))
          [CL::TYPES-WITH-DEFNS (TYPESOF CL::NAME CL::GIVEN-TYPES NIL (CL:IF (OR (CL:MEMBER :CURRENT CL::OPTIONS
                                                                                            )
                                                                               (CL:MEMBER 'CURRENT CL::OPTIONS
                                                                                            ))
                                                                           'CURRENT
                                                                           '?)
                                     #'(LAMBDA (X)
                                          (NEQ (GET X 'EDITDEF)
                                                'NILL]
          (CL::POSSIBLE-TYPES (COND
                                  ([AND (NULL CL::GIVEN-TYPES)
                                        (CL:SYMBOLP CL::NAME)
                                        (NOT (NULL *ED-OFFERS-PROPERTY-LIST*))
                                        (find X on (GETPROPLIST CL::NAME) by (CDDR X)
                                            suchthat (NULL (GET (CAR X)
                                                                'PROPTYPE]
```
                                   ;; if we're supposed to offer PROPERTY-LIST as an edit type, and this name has a property list with other
                                   ;; than system properties on it, then add IL:PROPERTY-LIST to the possible types.
```
                                   (CONS 'PROPERTY-LIST CL::TYPES-WITH-DEFNS))
                                  (T CL::TYPES-WITH-DEFNS)))
           (TYPE))
        (CL:WHEN (CL:MEMBER 'PROPERTY-LIST CL::OPTIONS)
```
            ;; this will allow PROPERTY-LIST to be specified as a fake filepkg type by the user (caller)
```
            (CL:SETQ CL::POSSIBLE-TYPES '(PROPERTY-LIST)))
        [CL:SETQ TYPE (if (CL:MEMBER :NEW CL::OPTIONS)
                          then
```
                               ;; if :NEW then install a blank definition first
```
                               (OR (INSTALL-PROTOTYPE-DEFN CL::NAME (OR CL::TYPES-WITH-DEFNS CL::GIVEN-TYPES)
                                       :NEW)
                                   (CL:RETURN-FROM ED NIL))
                          elseif (CDR CL::POSSIBLE-TYPES)
                          then
```
                               ;; Many types were found/given. Ask the user which to use.
```
                               (if CL::FROM-DISPLAY
                                   then (OR (MENU (create MENU
                                                      ITEMS _ CL::POSSIBLE-TYPES
                                                      TITLE _ (CL:FORMAT NIL "Edit which definition of ~S ?"
                                                                     CL::NAME)))
                                            (CL:RETURN-FROM ED NIL))
                                   else (ASKUSER NIL (CAR CL::POSSIBLE-TYPES)
```

```
                                          (CL:FORMAT NIL "Edit which ~A definition of ~S ? "
                                                  CL::POSSIBLE-TYPES CL::NAME)
                                          CL::POSSIBLE-TYPES))
                  elseif (NOT (NULL CL::POSSIBLE-TYPES))
                      then ;; Exactly one type was found.

                              (if CL::FROM-DISPLAY
                                  then                                         ; prepare the prompt window
                                       (TERPRI PROMPTWINDOW))
                              (CL:FORMAT (if CL::FROM-DISPLAY
                                             then PROMPTWINDOW
                                             else T)
                                    "Editing ~A ~A ~S.~%%"
                                    (CAR CL::POSSIBLE-TYPES)
                                    (CL:IF (EQ (CAR CL::POSSIBLE-TYPES)
                                               'PROPERTY-LIST)
                                        "of"
                                        "definition of")
                                    CL::NAME)
                              (CAR CL::POSSIBLE-TYPES)
                  else ;; No types were found.  Use the DefDefiner prototyping machinery.

                          (OR (INSTALL-PROTOTYPE-DEFN CL::NAME CL::GIVEN-TYPES)
                              (CL:RETURN-FROM ED NIL]
        (CL:IF (EQ TYPE 'PROPERTY-LIST)
            (EDITE (GETPROPLIST CL::NAME)
                   NIL CL::NAME 'PROPLST NIL CL::OPTIONS)
            (EDITDEF CL::NAME TYPE NIL NIL CL::OPTIONS))
        (CL:RETURN-FROM ED CL::NAME)))


(CL:DEFUN INSTALL-PROTOTYPE-DEFN (NAME &REST ARGS)

;;; Explain to the user that the given name has no definitions (of the given type, if any) and give them the chance to pick a new, dummy definition to
;;; install under that name.  Return the file-manager type of the definition installed.  If TYPES were supplied, the one returned should be one of them.  If
;;; no dummy was selected, return NIL.  If the third arg in :NEW then don't bother with the "no defn found" message, cause the user intends to intall
;;; anew.

    [CL:FLET
     [(MAKE-AND-INSTALL (TYPE DEFINER)
            (LET ((DFNFLG 'PROP))
                 (DECLARE (CL:SPECIAL DFNFLG))
                 (EVAL (XCL::MAKE-PROTOTYPE NAME TYPE DEFINER]
      (LET*
       ((DEFINER-HELP-STRING (CL:FORMAT NIL "Installs a definition for ~S using this definer." NAME))
        [REQUESTED-TYPES (AND (LISTP ARGS)
                              (MKLIST (CAR ARGS]
        (NEW-DEFN-FLG (AND (LISTP ARGS)
                           (EQ (CADR ARGS)
                               :NEW)))
        [TYPES-WITH-PROTOTYPES (IF (NULL REQUESTED-TYPES)
                                   THEN (XCL::PROTOTYPE-DEFN-TYPES)
                                   ELSE (INTERSECTION REQUESTED-TYPES (XCL::PROTOTYPE-DEFN-TYPES]
        PROTOTYPE-TYPE)
       (IF (AND NEW-DEFN-FLG TYPES-WITH-PROTOTYPES)
           THEN (IF (CDR TYPES-WITH-PROTOTYPES)
                    THEN (CL:FORMAT T "Installing new definition for ~S~%%" NAME)
                    ELSE (CL:FORMAT T "Installing new ~S definition for ~S~%%" (CAR TYPES-WITH-PROTOTYPES)
                                  NAME))
         ELSEIF (NULL REQUESTED-TYPES)
           THEN (CL:FORMAT T "~S has no definitions.~%%" NAME)
         ELSEIF (NULL (CDR REQUESTED-TYPES))
           THEN (CL:FORMAT T "~S has no ~A definition.~%%" NAME (CAR REQUESTED-TYPES))
         ELSE (CL:FORMAT T "~S has no definition of any of these types:~%%  ~A~%%" NAME REQUESTED-TYPES))
       [IF (NULL TYPES-WITH-PROTOTYPES)
           THEN (CL:RETURN-FROM INSTALL-PROTOTYPE-DEFN NIL)
         ELSEIF (NULL (CDR TYPES-WITH-PROTOTYPES))
           THEN (SETQ PROTOTYPE-TYPE (CAR TYPES-WITH-PROTOTYPES))
         ELSE
         (CL:FORMAT T "Select a type of dummy definition to install.~%%")
         (LET*
          ([MENU
            (CREATE MENU
                  TITLE _ "Select a type for a dummy defn:"
                  ITEMS _
                  (APPEND [FOR TYPE IN TYPES-WITH-PROTOTYPES
                              COLLECT `(,TYPE '(:TYPE ,TYPE)
                                               "Displays a menu of definers for this type."
                                               (SUBITEMS ,@(FOR DEFINER IN (XCL::PROTOTYPE-DEFINERS-FOR-TYPE TYPE)
                                                              COLLECT `(,DEFINER '(:DEFINER ,TYPE ,DEFINER)
                                                                                 ,DEFINER-HELP-STRING]
                         (LIST '("Don't make a dummy defn" NIL]
           (RESULT (MENU MENU)))
          (CL:ECASE (CL:FIRST RESULT)
              (:TYPE (SETQ PROTOTYPE-TYPE (CL:SECOND RESULT)))
              (:DEFINER
                (MAKE-AND-INSTALL (CL:SECOND RESULT)
```

```
                              (CL:THIRD RESULT))
                      (CL:RETURN-FROM INSTALL-PROTOTYPE-DEFN (CL:SECOND RESULT)))
                 ((NIL) (CL:RETURN-FROM INSTALL-PROTOTYPE-DEFN NIL)))]
        (CL:FORMAT T "Select a definer to use for a dummy definition.~%%")
        (LET [(DEFINER (MENU (CREATE MENU
                                     TITLE _ "Select a definer for a dummy defn:"
                                     ITEMS _ (APPEND (XCL::PROTOTYPE-DEFINERS-FOR-TYPE PROTOTYPE-TYPE)
                                                     (LIST '("Don't make a dummy defn" NIL]
             (IF DEFINER
                 THEN (MAKE-AND-INSTALL PROTOTYPE-TYPE DEFINER)
                      PROTOTYPE-TYPE
                 ELSE NIL])
```

(DEFINEQ

(**EDITDEF.FNS**
```
  [LAMBDA (NAME EDITCOMS OPTIONS)                                            ; Edited 20-Nov-87 14:25 by woz
    (PROG (DEF)
      LP  (COND
            ((EXPRP (SETQ DEF (OR (GET NAME 'ADVISED)
                                  (GET NAME 'BROKEN)
                                  NAME)))
             (EDITE (if (LITATOM DEF)
                        then (GETD DEF)
                        else DEF)
                    EDITCOMS NAME 'FNS NIL OPTIONS)
             (RETURN NAME))
            ((EXPRP (SETQ DEF (GETPROP NAME 'EXPR]

             ;; woz: don't use edit type PROP anymore.  Let putdef for fns worry about where the definition goes.

             (EDITE DEF EDITCOMS NAME 'FNS NIL OPTIONS)
             (RETURN NAME))
            ((EDITFB NAME)
             (GO LP))
            (T
             ;; Used to call EDITFERROR to check for MACROS definition or install dummy FNS defintion.  FNS can no longer be coerced to
             ;; MACROS, and the new prototype stuff handles the other case.  So if we're here, it's because EDITFB failed to find the definition,
             ;; and thus NAME is not editable.

             (CL:FORMAT *ERROR-OUTPUT* "Could not find fns definition for ~a." NAME)
             (ERROR "Could not find fns definition for " NAME T])
```

(**EDITF**
```
  [NLAMBDA EDITFX                                                            ; Edited 11-Jun-90 15:44 by jds
    (SETQ EDITFX (NLAMBDA.ARGS EDITFX))
    (LET ((FNTYPE 'FNS))
         (EDITDEF (if EDITFX
                      then (COND
                             ((HASDEF (CAR EDITFX)
                                      'FUNCTIONS
                                      '? EDITFX)
                              (SETQ FNTYPE 'FUNCTIONS)
                              (CAR EDITFX))
                             ((HASDEF (CAR EDITFX)
                                      'FNS
                                      '? EDITFX)
                              (CAR EDITFX)))
                      else (PROGN (PRIN1 "Editing " T)
                                  (PRINT LASTWORD T)))
                  FNTYPE NIL (CDR EDITFX])
```

(**EDITFB**
```
  [LAMBDA (FN)                                                               ; Edited 27-Jan-87 14:50 by Pavel
    (PROG (FL TEM)
          (COND
            ((SETQ FL (EDITLOADFNS? FN (AND (NULL EDITLOADFNSFLG)
                                            '"not editable, do you want to load it PROP from")
                                    (NULL EDITLOADFNSFLG)))
             (LOADFNS FN FL 'PROP))
            ((AND (EQ (NARGS 'WHEREIS)
                      4)
                  (SETQ FL (EDITLOADFNS? FN '"not editable; do you want to LOADFROM PROP the file" T T)))
             (LOADFROM FL (LIST FN)
                       'PROP))
            (T (RETURN)))
          (COND
            ((GETPROP FN 'EXPR)
             (RETURN T))
            (T (PRINTOUT T "** Not found on " FL T])
```

(**EDITFNS**
```
  [NLAMBDA X                                                                 (* DD%: " 7-Oct-81 20:56")

    ;; FNS is a list (or name of a list) of functions to be edited;  (CDR X) are the operations to be performed.

    (SETQ X (MKLIST X))
```

```
      (MAPC [COND
               ((LISTP (CAR X))
                (STKEVAL 'EDITFNS (CAR X)
                        NIL
                        'INTERNAL))
               (T                                                                          ; If (CAR X) is name of a file, do editfns on its functions.
                  (OR (LISTP (EVALV (CAR X
                                    'EDITFNS))
                      (AND (GETPROP (OR (AND DWIMFLG (MISSPELLED? (CAR X)
                                                               70 FILELST NIL X))
                                        (CAR X))
                                    'FILE)
                           (FILEFNSLST (CAR X)))
                      (STKEVAL 'EDITFNS (CAR X)
                              'INTERNAL]
            (FUNCTION (LAMBDA (Y)
                         (ERSETQ (APPLY 'EDITF (CONS (PROG1 (PRIN2 Y T T)
                                                           (SPACES 1 T))
                                                    (CDR X])
```

₍**EDITLOADFNS?**

```
  [LAMBDA (FN STR ASKFLG FILES)                                                      ; Edited 21-May-2024 18:18 by mth
                                                                                    (* lmm "20-Nov-86 21:23")

    ;; Value is name of file from which function or functions can be loaded.  If STR is non-NIL, user is asked to approve, and STR used in the message.
    ;; EDITLOADFNS?  is also used by prettyprint

    (AND FN FILEPKGFLG (PROG ((LST (WHEREIS FN 'FNS FILES))
                              FILE DATES FD)
                             (OR (COND
                                   ((EQ FILES T)

                                     ;; if FILES = T, means consult data base.  if user has removed a function from one of those files, as
                                     ;; evidenced by the fact that editloafns?  was called with files=T, then dont offer that file.

                                     (SETQ LST (LDIFFERENCE LST FILELST)))
                                   (T LST))
                                 (RETURN))
                             [SETQ FILE (COND
                                          ((CDR LST)
                                           (PRIN2 FN T)
                                           (MAPRINT LST T " is contained on " "
                                                   " " and ")
                                           (OR (ASKUSER NIL NIL "indicate which file to use: " (MAKEKEYLST
                                                                                                LST)
                                                        T)
                                               (RETURN)))
                                          (T (CAR LST]
                             [SETQ DATES (LISTP (GETPROP FILE 'FILEDATES]

                     ;; only look at file in FILEDATES if the file has been LOADed or LOADFROMd

                             (SETQ FILE (OR (AND DATES (FMEMB (CDAR (GETPROP FILE 'FILE))
                                                              '(LOADFNS T))
                                                       (INFILEP (CDAR DATES)))
                                            (FINDFILE FILE T)
                                            (RETURN)))
                             [COND
                               ((AND DATES (NEQ FILE (CDAR DATES)))
                                                                    ; found a different file than in FILEDATES

                                (COND
                                  ((EQUAL (CAAR DATES)
                                          (SETQ FD (FILEDATE FILE)))
                                                                    ; found a goood version of file on a different name.  smash name
                                   (/RPLACD (CAR DATES)
                                            FILE))
                                  (T (CL:FORMAT *TERMINAL-IO* "*** Note: loading definition from ~A dated
                                         ~A~&while file ~A dated ~A is the version currently loaded." FILE
                                         FD (CDAR DATES)
                                         (CAAR DATES]
                             (COND
                               ((STREQUAL STR ""))
                               ((NULL ASKFLG)
                                (if STR
                                    then (EXEC-FORMAT "~&~A~A~&" STR FILE)
                                  else (EXEC-FORMAT "~&Loading definition of ~S from ~A.~&" FN FILE)))
                               ((NEQ (ASKUSER DWIMWAIT 'Y (LIST FN STR FILE)
                                             NIL T)
                                     'Y)
                                (RETURN)))
                             (RETURN FILE])
```

₍**EDITMODE**

```
  [LAMBDA (NEWMODE)
```

```
;;; WOZ- 1/9/91.  Took (setq newmode (il:u-case newmode)) out because it puts newmode in the IL: package.  This doesn't work with my new definition
;;; of what an editor is, ie a symbol whos function cell can be applied to (structure props options).  The rest of the old code , eg Definition-for-EDITL
;;; remains until DEDIT goes away and the TTY/EDITOR becomes a programmatic structure editor only.
```

```
        (PROG1 (if (NOT (DISPLAYSTREAMP (TTYDISPLAYSTREAM)))
                  then  ;; not a display, always say teletype. This is mainly for chatserver

                     'TELETYPE
                else (SELECTQ *EDITMODE*
                        (DISPLAY *DISPLAY-EDITOR*)
                        *EDITMODE*))
            ;; return old value, and, if new value given, set it.

            (CL:WHEN NEWMODE
                (LET (new.display.editor)
                    (if (EQ NEWMODE 'STANDARD)
                        then                                          ; Obselete terminology
                            (SETQ NEWMODE 'TELETYPE)
                      elseif (AND (EQ NEWMODE 'DISPLAY)
                                  *DISPLAY-EDITOR*)
                        then (SETQ NEWMODE *DISPLAY-EDITOR*)
                      elseif (NEQ NEWMODE 'TELETYPE)
                        then                                          ; if we've been called with the name of a display editor, make it
                                                                      ; the default display editor
                            (SETQ new.display.editor NEWMODE))
                    (if (NOT (GETPROP NEWMODE 'Definition-for-EDITL))
                        then (CL:CERROR "Ignore call to EDITMODE" "~S is unrecognized argument to EDITMODE" NEWMODE
                                  )
                        else (if new.display.editor
                                 then (SETQ *DISPLAY-EDITOR* new.display.editor))
                            (SETQ *EDITMODE* NEWMODE)))))])
```

(**EDITP**
```
  [NLAMBDA EDITPX                                              (* lmm "10-Jun-85 17:12")
    (SETQ EDITPX (NLAMBDA.ARGS EDITPX))
    (PROG ((ATM (CAR EDITPX)))
          [COND
            ((AND DWIMFLG (NLISTP (GETPROPLIST ATM)))
             (SETQ ATM (OR (MISSPELLED? ATM 75 USERWORDS NIL NIL (FUNCTION GETPROPLIST))
                           ATM]
          (EDITE (GETPROPLIST ATM)
                 (CDR EDITPX)
                 ATM
                 'PROPLST)
          (RETURN ATM])
```

(**EDITV**
```
  [NLAMBDA EDITVX                                              (* lmm " 2-Sep-85 13:17")
    (SETQ EDITVX (NLAMBDA.ARGS EDITVX))
    (LET* [[VAR (OR (CAR EDITVX)
                    (PROGN (PRIN1 "= " T)
                           (PRINT LASTWORD T]
           (FRAME (AND VAR (STKSCAN VAR]
          (if FRAME
              then (EDITE (ENVEVAL VAR FRAME NIL T)
                         (CDR EDITVX)
                         VAR)
            elseif (SETQ VAR (HASDEF VAR 'VARS 'CURRENT T))
              then (EDITDEF VAR 'VARS 'CURRENT (CDR EDITVX))
            else (ERROR VAR "not editable"])
```

(**DC**
```
  [NLAMBDA FILE                                                ; Edited 18-Mar-87 16:03 by woz
                                                               ; Edits COMS of file FILE
    (LET ((*EDITMODE* 'DISPLAY)
          (ARGS (NLAMBDA.ARGS FILE)))
        ;; (APPLY (QUOTE EDITV) (FILECOMS (OR (HASDEF (CAR (NLAMBDA.ARGS FILE)) (QUOTE FILE) NIL T) (ERROR FILE 'is not a
        ;; loaded file' T))))

        (if (LISTP ARGS)
            then (SETQ *LAST-DC* (CAR ARGS)))
        (if *LAST-DC*
            then (ED *LAST-DC* '(FILES :DONTWAIT))
          else (ERROR "No saved file name." "Edit aborted."])
```

(**DF**
```
  [NLAMBDA FN                                                  ; Edited 18-Mar-87 17:00 by woz
    (LET ((*EDITMODE* 'DISPLAY)
          (ARGS (NLAMBDA.ARGS FN)))
        ;; (APPLY (COND ((EQ (CADR (LISTP FN)) (QUOTE NEW)) (QUOTE EDITFERROR)) (T (QUOTE EDITF))) (NLAMBDA.ARGS FN))

        ;; DF used to look for MACROS under EDITFERROR.  Decided this is bad, because could get the macro without noticing the fns if the file
        ;; was sysloaded.  Now just look for FUNCTIONS and FNS.

        (if (LISTP ARGS)
            then (SETQ *LAST-DF* (CAR ARGS)))
        (if *LAST-DF*
```

```
                then [ED *LAST-DF* (if (AND (CDR ARGS)
                                           (EQ (CADR ARGS)
                                               'NEW))
                                      then '(FUNCTIONS FNS :DONTWAIT :NEW)
                                      else '(FUNCTIONS FNS :DONTWAIT]
              else (ERROR "No saved function name." "Edit aborted.")]
```

(**DP**
```
  [NLAMBDA ATOM                                               ; Edited 18-Mar-87 16:16 by woz
     (LET ((*EDITMODE* 'DISPLAY)
           (ARGS (NLAMBDA.ARGS ATOM)))                        ; (APPLY (QUOTE EDITPROP) (NLAMBDA.ARGS ATOM))
          [if (LISTP ARGS)
              then (if (CDR ARGS)
                       then                                   ; specific PROP to edit. remember (ATOM PROP)
                            (SETQ *LAST-DP* ARGS)
                       else                                   ; edit whole plist. remember ATOM
                            (SETQ *LAST-DP* (CAR ARGS]
          (if *LAST-DP*
              then [ED *LAST-DP* (if (LISTP *LAST-DP*)
                                     then '(PROPS :DONTWAIT)
                                     else '(PROPERTY-LIST :DONTWAIT]
              else (ERROR "No saved symbol name." "Edit aborted."])
```

(**DV**
```
  [NLAMBDA VAR                                                ; Edited 18-Mar-87 12:43 by woz
     (LET ((*EDITMODE* 'DISPLAY)
           (ARGS (NLAMBDA.ARGS VAR)))                         ; (APPLY (QUOTE EDITV) (NLAMBDA.ARGS VAR))
          (if (LISTP ARGS)
              then (SETQ *LAST-DV* (CAR ARGS)))
          (if *LAST-DV*
              then (ED *LAST-DV* '(VARIABLES VARS :DONTWAIT))
              else (ERROR "No saved variable name." "Edit aborted."])
```

(**EDITPROP**
```
  [LAMBDA (NAME PROP)                                         (* bas%: "21-MAR-83 20:29")
     (COND
        (PROP (EDITDEF (LIST NAME PROP)
                       'PROPS))
        (T (APPLY 'EDITP NAME)))
```

(**EF**
```
  [NLAMBDA FN                                                 (* jow "16-Oct-86 11:41")
     (LET ((*EDITMODE* 'TELETYPE))
          (APPLY (COND
                    ((EQ (CADR (LISTP FN))
                         'NEW)
                     'EDITFERROR)
                    (T 'EDITF))
                 (NLAMBDA.ARGS FN])
```

(**EP**
```
  [NLAMBDA ATOM                                               (* jow "16-Oct-86 11:42")
     (LET ((*EDITMODE* 'TELETYPE))
          (APPLY 'EDITPROP (NLAMBDA.ARGS ATOM])
```

(**EV**
```
  [NLAMBDA VAR                                                (* jow "16-Oct-86 11:42")
     (LET ((*EDITMODE* 'TELETYPE))
          (APPLY 'EDITV (NLAMBDA.ARGS VAR])
```

(**EDITE**
```
  [LAMBDA (EXPR COMS ATM TYPE IFCHANGEDFN OPTIONS)            ; Edited 17-Mar-87 11:59 by woz
     (if (AND EXPR (NLISTP EXPR)
              (SELECTQ (EDITMODE)
                  (TELETYPE                                   ; don't inspect
                         NIL)
                  (SEDIT (NOT (STRINGP EXPR)))                ; Sedit can't handle strings, it returns the old string
                         T)
                  T))
         then  ;; this used to be done by redefining EDITE on the file INSPECT. Its not clear that it is still a good idea

              (INSPECT EXPR)
         else (CL:FUNCALL (GET (EDITMODE)
                               'Definition-for-EDITE)
                     EXPR COMS ATM TYPE IFCHANGEDFN OPTIONS])
```

(**EDITL**
```
  [LAMBDA (L COMS ATM MESS EDITCHANGES)                       (* lmm "12-Nov-86 15:18")
     (CL:FUNCALL (GET (EDITMODE)
```

```
                          'Definition-for-EDITL)
            L COMS ATM MESS EDITCHANGES])
```

)

;; Time stamp on functions when edited

;; User enables this by an (ADDTOVAR INITIALSLIST (USERNAME)) in his INIT.LISP. E.g. (ADDTOVAR INITIALSLIST (MASINTER)) The date fixup
;; is enabled by the variable INITIALS. The function SETINITIALS sets INITIALS from INITIALSLIST and USERNAME at load time, and after a sysin.

(DEFINEQ

(**NEW/EDITDATE**

```
  [LAMBDA (OLDATE INITLS)                                          (* lmm "12-Nov-86 15:18")
    (CL:FUNCALL (GET (EDITMODE)
                      'Definition-for-EDITDATE)
           OLDATE INITLS])
```


(**FIXEDITDATE**

```
  [LAMBDA (EXPR)
```

   ;; Edited 22-Jun-2022 13:31 by rmk

   ;; Edited 13-May-2022 08:11 by rmk

   ;; Edited 8-May-2022 22:49 by rmk

   ;; Edited 19-Jan-2022 23:08 by rmk

   ;; Edited 8-Dec-2021 16:11 by rmk: Updated to add dates to the initial undated comments that begins with current-editor initials, to provide a kind
   ;; of dated change-log capability.

   ;; Edited 27-Sep-2018 22:04 by rmk:

   ;; Edited 31-Mar-2000 17:13 by rmk:

   ;; Edited 17-Jul-89 11:13 by jtm: ; 18-JUL-78 21:11

   ;; Inserts or replaces previous edit date. This retains multiple edits (at least one day apart or by different editor) unless
   ;; *REPLACE-OLD-EDIT-DATES*. Note that the new date doesn't show up within the current SEDIT session, you have to exit and re-edit to see it.

```
    (CL:WHEN (AND INITIALS (LISTP EXPR)
                  (LISTP (CDR EXPR)))
        (PROG (E)
```

     ;; Normalize out the colon, add it back if needed.

```
              (COND
                 ((FMEMB (CAR EXPR)
                         LAMBDASPLST)
```

      ;; insert the edit date after the argument list

```
                  (SETQ E (CDDR EXPR)))
                 [(FMEMB (GETPROP (CAR EXPR)
                          ':DEFINER-FOR)
                         EDITDATE-ARGLIST-DEFINERS)
```

      ;; insert the edit date after the argument list

```
                  (SETQ E (CDDR EXPR))
                  (while (NOT (CL:LISTP (CAR E))) do (SETQ E (CDR E)) finally (SETQ E (CDR E]
                 ((FMEMB (GETPROP (CAR EXPR)
                          ':DEFINER- FOR)
                         EDITDATE-NAME-DEFINERS)
```

      ;; insert the edit date after the name

```
                  (SETQ E (CDDR EXPR)))
                 (T (RETURN)))
           RETRY
              [COND
                 ((NLISTP E)
                  (RETURN))
                 ((LISTP (CAR E))
                  (SELECTQ (CAAR E)
                      ((CLISP%: DECLARE)
                       (SETQ E (CDR E))
                       (GO RETRY))
                      (BREAK1 (COND
                                ((EQ (CAR (CADAR E))
                                     'PROGN)
                                 (SETQ E (CDR (CADAR E)))
                                 (GO RETRY))))
                      (ADV-PROG                                     ; No easy way to mark cleanly the date of an advised function
                             (RETURN))
                      (COND
                        ((AND (EQ (CAAR E)
                                  COMMENTFLG)
                              (EQ (CADAR E)
                                  'DECLARATIONS%:))
                         (SETQ E (CDR E))
                         (GO RETRY]
```

     ;; E is now the cell that the date will attach to or whose CAR will be updated.

```
                    [LET (PARSE COMMENTLEVEL ENDINITIALS (INITLS (CL:IF (EQ (CHARCODE %:)
                                                                         (NTHCHARCODE INITIALS -1))
                                                              (SUBSTRING INITIALS 1 -2)
                                                              INITIALS)))
              (IF *REPLACE-OLD-EDIT-DATES*
                  THEN  ;; Strip out all previous modern-format edit dates. Since EDITDATE? only recognizes that format, hand editing is
                        ;; needed if prehistoric dates are really not desired. We don't strip out anything with a further comment.
                      (BIND (TAIL _ E) WHILE (AND (LISTP TAIL)
                                                  (EDITDATE? (CAR TAIL)))
                         DO (SETQ TAIL (CDR TAIL)) FINALLY (CL:UNLESS (EQ E TAIL)
                                                                      (/RPLACD E TAIL)))
                   ;; Now (CAR E) may or may not be a (no-REST) editdate, but there are none afterwards.
                      (IF (SETQ PARSE (EDITDATE? (CAR E)
                                                 T))
                          THEN                                              ; Smash it
                              (EDITDATE (CAR E)
                                        INITLS
                                        (CADDR PARSE))
                        ELSE (/ATTACH (EDITDATE NIL INITLS)
                                      E))
                ELSE (IF (SETQ PARSE (EDITDATE? (CAR E)
                                                T))
                         THEN ;; If edited by the same editor within a day, then update the previous timestamp rather than cluttering
                              ;; with a new one.  Presumably this is the next event in the same longer editing session, and it avoids
                              ;; stacking up uninformative dates from in-and-out editing during a session.
                             (IF [AND (STRING.EQUAL INITLS (CADR PARSE))
                                     (ILEQ (IDIFFERENCE (IDATE)
                                                        (CAR PARSE))
                                           (CONSTANT (TIMES 24 3600]
                                 THEN ;; Same edit session with the same author: update the last previous timestamp.
                                     (/RPLACA E (EDITDATE (CAR E)
                                                          INITLS
                                                          (CADDR PARSE)))
                               ELSE ;; Different edit sequence, attach a new timestamp in front of any old ones, without the rest
                                   (/ATTACH (EDITDATE NIL INITLS)
                                            E))
                       ELSEIF (AND [EQ COMMENTFLG (CAR (LISTP (CAR E]
                                   (MEMB [CAR (LISTP (SETQ PARSE (CDAR E]
                                         '(; ;; ;;;))
                                   [STRINGP (SETQ PARSE (CAR (LISTP (CDR PARSE]
                                   (SETQ ENDINITIALS (STRPOS INITLS PARSE 1 NIL NIL T UPPERCASEARRAY)))
                          THEN ;; Just an ordinary comment in first position, with initials: in front.  Upgrade it to an edit date.
                              (SETQ PARSE (CONCAT INITLS (CL:IF (EQ (CHARCODE %:)
                                                                   (NTHCHARCODE PARSE ENDINITIALS))
                                                               ""
                                                               ": ")
                                                  (SUBSTRING PARSE ENDINITIALS)))
                              (/RPLACA E (EDITDATE (CAR E)
                                                   NIL PARSE))
                         ELSE ;; First edit: we didn't see an old date to compare with or smash, not even an initials: xxx form.
                             (/ATTACH (EDITDATE NIL INITLS)
                                      E))
              ;; Make sure that all the previous dates have the same comment level.
              [SETQ COMMENTLEVEL (CADR (FOR C CFLAG IN E WHILE (EDITDATE? C)
                                         LARGEST (NCHARS (CADR C]
              (FOR C IN E WHILE (EDITDATE? C) UNLESS (EQ (CADR C)
                                                        COMMENTLEVEL)
                 DO (/RPLACA (CDR C)
                             COMMENTLEVEL]
              (RETURN EXPR)))])
```

(**EDITDATE?**
```
  [LAMBDA (COMMENT PARSE)                                   ; Edited 13-May-2022 08:05 by rmk
                                                            ; Edited  6-May-2022 23:39 by rmk
```
    ;; This determines whether this is a dated or initialed comment that is potentially reusable in the current context. It recognizes comments with
    ;; edit-date strings of the following formats:

;;;   "Edited <date> by <initials>(:)"

;;;   "Edited <date> by <initials>: <comment>"

;;;; Value is NIL if the comment is not in one of these formats. Otherwise, if PARSE, then the value is a list (<date> <initials> <comment>), else T.

;;;

;;;; The caller can compare against current time and current-user initials to decide whether to smash or add.

;;;; There is no harm in not recognizing prehistoric formats, new dates will always be added on.

```
(LET ((TAIL COMMENT)
       STRING BYPOS (IPOS 1)
      DATE I IENDPOS REST)
     (CL:WHEN [AND (EQ COMMENTFLG (CAR (LISTP TAIL)))
                   (MEMB [CAR (LISTP (SETQ TAIL (CDR TAIL]
                         '(; ;; ;;;))
                   (STRINGP (SETQ STRING (CAR (SETQ TAIL (CDR TAIL]
          (SETQ STRING (CL:STRING-TRIM '(#\Space)
                                       STRING))
          (CL:WHEN [AND [STREQUAL "Edited " (SUBSTRING STRING 1 7 (CONSTANT (CONCAT]
                        (SETQ BYPOS (STRPOS " by " STRING 8))
                        (SETQ DATE (IDATE (SUBSTRING STRING 8 (SUB1 BYPOS]
```

;; Standard format, initials should be next.

```
               (SETQ IPOS (IPLUS BYPOS 4))
```

;; The next substring may be the initials, or it may be a comment string without initials prepended. We pull off the next substring
;; and strip the colon if any, but also return the whole trailing string.

```
               (CL:WHEN (IGREATERP (NCHARS STRING)
                                   IPOS)
                   [SETQ IENDPOS (SUB1 (OR (STRPOS " " STRING IPOS)
                                           (ADD1 (NCHARS STRING]
                   (SETQ I (SUBSTRING STRING IPOS IENDPOS))
                   (CL:WHEN (ILESSP (NCHARS I)
                                    12)                           ; Sanity check: Initials should be short.
                       (CL:WHEN (EQ (CHARCODE %:)
                                    (NTHCHARCODE I -1))           ; Normalize out the colon
                           (SETQ I (SUBSTRING I 1 -2)))
                       (CL:WHEN (NTHCHARCODE STRING (ADD1 IENDPOS))
                                                                  ; At least one  REST character
                           (SETQ REST (SUBSTRING STRING IPOS)))
                       (OR (NOT PARSE)
                           (LIST DATE I REST)))))))])
```

⟨**EDITDATE**

```
  [LAMBDA (OLDDATE INITLS REST)
```

;; Edited 6-May-2022 23:13 by rmk: If REST is non-NIL, assumes that it already has the correct INITLS packed onto the font.

;; Edited 8-Dec-2021 17:58 by rmk: Upgraded to make sure that the comment includes REST

;; 20-Nov-86 23:23 by Masinter

;; Generates a new date from an old one. Packs : onto INITLS if there is a REST. In the REST case we upgrade a singe semicolon to a double.

```
    (LET ((EDITSTRING (CONCAT "Edited " (DATE (DATEFORMAT NO.SECONDS))
                              " by "
                              (CL:IF REST
                                  ""
                                  INITLS)))
          NEWDATE OLDSEMI)
         (CL:WHEN REST
             (SETQ EDITSTRING (CONCAT EDITSTRING REST)))
         (CL:WHEN OLDDATE
             (SETQ OLDSEMI (CADR OLDDATE)))
         (SETQ NEWDATE (LIST (CL:IF REST
                                 (OR OLDSEMI ';;)
                                 ';)
                             EDITSTRING))
         (COND
            ((EQMEMB (CAR (LISTP OLDDATE))
                     COMMENTFLG)
```

 ;; Destructively alter old date. This is for benefit of DEDIT being able to find the old form to reprint

```
             (/RPLACD OLDDATE NEWDATE))
            (T (CONS (OR (CAR (LISTP COMMENTFLG))
                         COMMENTFLG)
                     NEWDATE])
```

⟨**SETINITIALS**

```
  [LAMBDA NIL                                                    ; Edited 20-Nov-86 23:22 by MASINTER
```

;;; Sets the default initials to appear in edited code, and sets the user's first name. Called following GREET via POSTGREETFORMS.

```
    (LET ((DFNFLG T)
          (FILEPKGFLG NIL)
          (USER (USERNAME NIL NIL T))
          POS TRIPLE)
         (SETQ POS (STRPOS "." USER))                            ; Find out if there's a period in the name, which would indicate
                                                                 ; that there's a registry on the end.
         [COND
            ((AND POS DEFAULTREGISTRY (STRING-EQUAL (SUBSTRING USER (ADD1 POS)
                                                               -1)
                                                    DEFAULTREGISTRY))
```

         ;; If there's a registry on the end, and it's the default registry, remove the registry. We assume that DEFAULTREGISTRY has been set
         ;; by the time that GREET has finished.

```
                (SETQ USER (SUBSTRING USER 1 (SUB1 POS]
            (COND
               [(find old TRIPLE in INITIALSLST suchthat (STRING-EQUAL USER (CAR TRIPLE)))
                                                                 ; OK we found his last name on the INITIALSLST.  Now break
                                                                 ; out his initials and first name
                (COND
                   ((NLISTP (CDR TRIPLE))                         ; old style
                    (SAVESET 'INITIALS (CDR TRIPLE)))
                   (T (SAVESET 'FIRSTNAME (CADR TRIPLE))
                      (SAVESET 'INITIALS (CADDR TRIPLE]
               (T (SAVESET 'INITIALS (COND
                                        ((NOT DEFAULTINITIALS)
                                         NIL)
                                        ((NEQ DEFAULTINITIALS T)
                                         DEFAULTINITIALS)
                                        (T USER])
)

(RPAQ? INITIALS )

(RPAQ? INITIALSLST )

(RPAQ? DEFAULTINITIALS T)

(RPAQ? *REPLACE-OLD-EDIT-DATES* NIL)

(MOVD? 'EDITDATE 'TTY/EDITDATE)
```

;; Moved from FILEPKG

```
(RPAQQ EDITDATE-ARGLIST-DEFINERS (FUNCTIONS TYPES))

(RPAQQ EDITDATE-NAME-DEFINERS (STRUCTURES VARIABLES))

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS EDITDATE-ARGLIST-DEFINERS EDITDATE-NAME-DEFINERS)
)

(RPAQ? COMMON-SOURCE-MANAGER-TYPES '(FUNCTIONS VARIABLES STRUCTURES TYPES SETFS OPTIMIZERS))

(PUTPROPS EDITINTERFACE FILETYPE CL:COMPILE-FILE)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA EV EP EF DV DP DF DC EDITV EDITP EDITFNS EDITF)

(ADDTOVAR NLAML )

(ADDTOVAR LAMA )
)

(PUTPROPS EDITINTERFACE COPYRIGHT ("Venue & Xerox Corporation" 1986 1987 1988 1990 1991 2024))
```

FUNCTION INDEX

VARIABLE INDEX

PROPERTY INDEX