

File created: 6-Jan-92 15:12:26 {DSK}<usr>local>lde>lispcore>sources>ADVISE.;

changes to: (IL:FUNCTIONS XCL:ADVISE-FUNCTION XCL:UNADVISE-FUNCTION XCL:READWISE-FUNCTION FINISH-ADVISING)

previous date: 16-May-90 11:55:52 {DSK}<usr>local>lde>lispcore>sources>ADVISE.;

Read Table: XCL

Package: SYSTEM

Format: XCCS

; Copyright (c) 1978, 1984, 1986, 1987, 1988, 1990, 1992 by Venue & Xerox Corporation. All rights reserved.  
; The following program was created in 1978 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.

```
(IL:RPAQQ IL:ADVISECOMS
  ((IL:STRUCTURES ADVICE)
   (IL:VARIABLES IL:ADVISED-FNS *UNADVISED-FNS*)
   ;;
   ;; Interlisp entry points.
   (IL:FNS IL:ADVISE IL:UNADVISE IL:READWISE)
   (IL:PROP IL:ARGNAMES IL:ADVISE)
   ;;
   ;; XCL entry points.
   (IL:FUNCTIONS XCL:ADVISE-FUNCTION XCL:UNADVISE-FUNCTION XCL:READWISE-FUNCTION)
   (IL:FUNCTIONS UNADVISE-FROM-RESTORE-CALLS FINISH-ADVISING FINISH-UNADVISING)
   ;;
   ;; The advice database.
   (IL:VARIABLES *ADVICE-HASH-TABLE*)
   (IL:FUNCTIONS ADD-ADVICE DELETE-ADVICE GET-ADVICE-MIDDLE-MAN SET-ADVICE-MIDDLE-MAN INSERT-ADVICE-FORM
    )
   (IL:SETFS GET-ADVICE-MIDDLE-MAN)
   ;;
   ;; Hacking the actual advice forms.
   (IL:FUNCTIONS CREATE-ADVISED-DEFINITION MAKE-AROUND-BODY)
   ;;
   ;; Dealing with the File Manager
   (IL:FILEPKGCOMS IL:ADVICE IL:ADVISE)
   (IL:FUNCTIONS XCL:REINSTALL-ADVICE)
   (IL:FUNCTIONS ADVICE-GETDEF ADVICE-PUTDEF ADVICE-DELDEF ADVICE-HASDEF ADVICE-NEWCOM
    ADVICE-FILE-DEFINITIONS ADVICE-CONTENTS ADVICE-ADDTOCOM)
   (IL:PROP IL:PROPTYPE IL:ADVISED)
   ;;
   ;; Dealing with old-style advice
   (IL:FUNCTIONS IL:READWISE1 ADD-OLD-STYLE-ADVICE CANONICALIZE-ADVICE-SYMBOL
    CANONICALIZE-ADVICE-WHEN-SPEC CANONICALIZE-ADVICE-WHERE-SPEC)
   (IL:DEFINE-TYPES XCL:ADVISED-FUNCTIONS)
   (IL:FUNCTIONS XCL:DEFADVICE)
   ;; Arrange for the proper package. Because of the DEFSTRUCT above, we must have the file dumped in the SYSTEM package.
   (IL:PROP (IL:MAKEFILE-ENVIRONMENT IL:FILETYPE)
    IL:ADVISE)
   (IL:DECLARE\ : IL:DONTEVAL@LOAD IL:DOEVAL@COMPILE IL:DONTCOPY IL:COMPILEVAR (IL:ADDVARS (IL:NLAMA
    IL:READWISE
    IL:UNADVISE
    )
    (IL:NLAML)
    (IL:LAMA
    IL:ADVISE)
    )))

(DEFSTRUCT (ADVICE (:TYPE LIST))
  BEFORE
  AFTER
  AROUND)

(DEFVAR IL:ADVISED-FNS NIL)

(DEFVAR *UNADVISED-FNS* NIL)

;;
;; Interlisp entry points.
```

(IL:DEFINEQ

(IL:ADVISE

(IL:LAMBDA IL:ARGS

; Edited 6-Apr-87 18:00 by Pavel

::: ADVISE the FN given. ADVISE1 is for advice of the type (foo IN bar)

(LET (IL:FN IL:WHEN IL:WHERE IL:WHAT)

:: First we straighten out the arguments given to us

(IL:SETQ IL:FN (IL:ARG IL:ARGS 1))

(CASE IL:ARGS

(2 (IL:SETQ IL:WHAT (IL:ARG IL:ARGS 2)))

(3 (IL:SETQ IL:WHEN (IL:ARG IL:ARGS 2))

(IL:SETQ IL:WHAT (IL:ARG IL:ARGS 3)))

(4 (IL:SETQ IL:WHEN (IL:ARG IL:ARGS 2))

(IL:SETQ IL:WHERE (IL:ARG IL:ARGS 3))

(IL:SETQ IL:WHAT (IL:ARG IL:ARGS 4)))

(T (IL:IF (< IL:ARGS 2)

IL:THEN (ERROR 'IL:TOO-FEW-ARGUMENTS :CALLEE 'IL:ADVISE :ACTUAL IL:ARGS :MINIMUM 2)

IL:ELSE (ERROR 'IL:TOO-MANY-ARGUMENTS :CALLEE 'IL:ADVISE :ACTUAL IL:ARGS :MAXIMUM 4))))

(IL:SETQ IL:WHEN (CANONICALIZE-ADVICE-WHEN-SPEC IL:WHEN))

(IL:SETQ IL:WHERE (CANONICALIZE-ADVICE-WHERE-SPEC IL:WHERE))

(IL:IF (IL:NLISTP IL:FN)

IL:THEN (XCL:ADVISE-FUNCTION IL:FN IL:WHAT :WHEN IL:WHEN :PRIORITY IL:WHERE)

IL:ELSEIF (IL:STRING-EQUAL (CADR IL:FN)

"IN")

IL:THEN (XCL:ADVISE-FUNCTION (FIRST IL:FN)

IL:WHAT :IN (THIRD IL:FN)

:WHEN IL:WHEN :PRIORITY IL:WHERE)

IL:ELSE (IL:FOR IL:X IL:IN IL:FN IL:JOIN (IL:IF (IL:NLISTP IL:X)

IL:THEN (XCL:ADVISE-FUNCTION IL:X IL:WHAT :WHEN IL:WHEN

:PRIORITY IL:WHERE)

IL:ELSE (XCL:ADVISE-FUNCTION (FIRST IL:X)

IL:WHAT :IN (THIRD IL:X)

:WHEN IL:WHEN :PRIORITY IL:WHERE))))))

(IL:UNADVISE

(IL:NLAMBDA IL:FNS

; Edited 6-Apr-87 16:21 by Pavel

(IL:SETQ IL:FNS (IL:NLAMBDA.ARGs IL:FNS))

(FLET ((IL:UNADVISE-ENTRY (IL:ENTRY)

(IL:IF (IL:LISTP IL:ENTRY)

IL:THEN (XCL:UNADVISE-FUNCTION (FIRST IL:ENTRY)

:IN

(THIRD IL:ENTRY))

IL:ELSE (XCL:UNADVISE-FUNCTION IL:ENTRY))))

(COND

((NULL IL:FNS)

(IL:FOR IL:ENTRY IL:IN (IL:REVERSE IL:ADVISED-FNS) IL:JOIN (IL:UNADVISE-ENTRY IL:ENTRY)))

((IL:EQUAL IL:FNS '(T))

(AND (NOT (NULL IL:ADVISED-FNS))

(IL:UNADVISE-ENTRY (CAR IL:ADVISED-FNS))))

(T (IL:FOR IL:ENTRY IL:IN IL:FNS IL:JOIN (IL:UNADVISE-ENTRY IL:ENTRY))))))

(IL:READVISE

(IL:NLAMBDA IL:FNS

; Edited 6-Apr-87 16:52 by Pavel

(IL:SETQ IL:FNS (IL:NLAMBDA.ARGs IL:FNS))

(FLET ((IL:READVISE-ENTRY (IL:ENTRY)

(IL:IF (IL:LISTP IL:ENTRY)

IL:THEN (XCL:READVISE-FUNCTION (FIRST IL:ENTRY)

:IN

(THIRD IL:ENTRY))

IL:ELSE (XCL:READVISE-FUNCTION IL:ENTRY))))

(COND

((NULL IL:FNS)

(IL:FOR IL:ENTRY IL:IN (IL:REVERSE \*UNADVISED-FNS\*) IL:JOIN (IL:READVISE-ENTRY IL:ENTRY)))

((IL:EQUAL IL:FNS '(T))

(AND (NOT (NULL \*UNADVISED-FNS\*))

(IL:READVISE-ENTRY (CAR \*UNADVISED-FNS\*))))

(T

; advise them all, in reverse order.

IL:JOIN (IL:READVISE-ENTRY IL:ENTRY)))

; simple case, advise just the last one that was unadvised.

; they gave us some functions, so advise THEM. We can't use

; READVISE-ENTRY here, because we may have to deal with

; old-style advice.

(IL:FOR IL:ENTRY IL:IN IL:FNS IL:JOIN (IL:READVISE1 IL:ENTRY))))))

)

(IL:PUTPROPS IL:ADVISE IL:ARGNAMES (IL:WHO IL:WHEN IL:WHERE IL:WHAT))

::

:: XCL entry points.

(DEFUN XCL:ADVISE-FUNCTION (XCL::FN-TO-ADVISE XCL::FORM &KEY (:IN XCL::IN-FN))

```

(WHEN :BEFORE)
(XCL::PRIORITY :LAST))
(MULTIPLE-VALUE-BIND (XCL::EXECUTABLE-TO-ADVISE XCL::NO-IN-FN)
(XCL::NAME-OF-EXECUTABLE XCL::FN-TO-ADVISE)
(COND
  ((AND (CONSP XCL::FN-TO-ADVISE)
        (NOT XCL::EXECUTABLE-TO-ADVISE))
   (IL:FOR XCL::FN IL:IN XCL::FN-TO-ADVISE IL:JOIN (XCL:ADVISE-FUNCTION XCL::FN XCL::FORM :IN XCL::IN-FN
:WHEN WHEN :PRIORITY XCL::PRIORITY)))
  ((AND (CONSP XCL::IN-FN)
        (NOT (XCL::NAME-OF-EXECUTABLE XCL::IN-FN)))
   (IL:FOR XCL::FN IL:IN XCL::IN-FN IL:JOIN (XCL:ADVISE-FUNCTION XCL::FN-TO-ADVISE XCL::FORM :IN XCL::FN
:WHEN WHEN :PRIORITY XCL::PRIORITY)))
  (T (LET (XCL::EXECUTABLE-TO-ADVISE-IN)
        (COND
          ((NULL XCL::FORM)
           (FORMAT *ERROR-OUTPUT* "No advice given, so nothing done."
            NIL)
           (IL:UNSAFE.TO.MODIFY XCL::FN-TO-ADVISE "advise")
           (FORMAT *ERROR-OUTPUT* "~S not advised.~%" XCL::FN-TO-ADVISE)
            NIL)
          (T (COND
              (XCL::IN-FN (SETQ XCL::EXECUTABLE-TO-ADVISE-IN (XCL::NAME-OF-EXECUTABLE XCL::IN-FN))
                (IF (NOT (HAS-CALLS XCL::EXECUTABLE-TO-ADVISE-IN XCL::EXECUTABLE-TO-ADVISE))
                    (ERROR "~S is not called from ~S." XCL::FN-TO-ADVISE XCL::IN-FN)))
              (T (IF (NULL (IL:GETD XCL::EXECUTABLE-TO-ADVISE))
                    (ERROR 'XCL:UNDEFINED-FUNCTION :NAME XCL::FN-TO-ADVISE))))
              (XCL:UNBREAK-FUNCTION XCL::FN-TO-ADVISE :IN XCL::IN-FN :NO-ERROR T)
              (COND
                ((NULL XCL::IN-FN)
                 ;; Adjust the database of advice for this function.
                 (WHEN (NOT (MEMBER XCL::FN-TO-ADVISE IL:ADVISEDFNS :TEST 'EQ))
                     ; If FN-TO-ADVISE is not currently advised, the new advice
                     ; replaces any that may have been given before.
                     (DELETE-ADVICE XCL::FN-TO-ADVISE))
                  (ADD-ADVICE XCL::FN-TO-ADVISE WHEN XCL::PRIORITY XCL::FORM)
                  ;; Finish off the process. This part is shared with READVISE-FUNCTION.
                  (FINISH-ADVISING XCL::FN-TO-ADVISE XCL::EXECUTABLE-TO-ADVISE))
                (T (LET* ((XCL::ADVICE-NAME ` (,XCL::FN-TO-ADVISE :IN ,XCL::IN-FN))
                        (XCL::ALREADY-ADVISED? (MEMBER XCL::ADVICE-NAME IL:ADVISEDFNS :TEST
'EQUAL)))
                    ;; Adjust the database of advice for this request.
                    (WHEN (NOT XCL::ALREADY-ADVISED?)
                        ; If not currently advised, the new advice replaces any that may
                        ; have been given before.
                        (DELETE-ADVICE XCL::ADVICE-NAME))
                      (ADD-ADVICE XCL::ADVICE-NAME WHEN XCL::PRIORITY XCL::FORM)
                      ;; Finish off the process. This part is shared with READVISE-FUNCTION.
                      (FINISH-ADVISING XCL::FN-TO-ADVISE XCL::EXECUTABLE-TO-ADVISE XCL::IN-FN
XCL::EXECUTABLE-TO-ADVISE-IN))))))))))

```

```

(DEFUN XCL:UNADVISE-FUNCTION (XCL::FN-TO-UNADVISE &KEY ((:IN XCL::IN-FN)
XCL::NO-ERROR)
(MULTIPLE-VALUE-BIND (XCL::EXECUTABLE-TO-UNADVISE XCL::NO-IN-FN)
(XCL::NAME-OF-EXECUTABLE XCL::FN-TO-UNADVISE)
(COND
  ((AND (CONSP XCL::FN-TO-UNADVISE)
        (NOT XCL::EXECUTABLE-TO-UNADVISE))
   (IL:FOR XCL::FN IL:IN XCL::FN-TO-UNADVISE IL:JOIN (XCL:UNADVISE-FUNCTION XCL::FN :IN XCL::IN-FN)))
  ((AND (CONSP XCL::IN-FN)
        (NOT (XCL::NAME-OF-EXECUTABLE XCL::IN-FN)))
   (IL:FOR XCL::FN IL:IN XCL::IN-FN IL:JOIN (XCL:UNADVISE-FUNCTION XCL::FN-TO-UNADVISE :IN XCL::FN)))
  (T (XCL:UNBREAK-FUNCTION XCL::FN-TO-UNADVISE :IN XCL::IN-FN :NO-ERROR T)
    (IF (NULL XCL::IN-FN)
        (LET ((XCL::ORIGINAL (GET XCL::EXECUTABLE-TO-UNADVISE 'IL:ADVISED)))
            (COND
              ((NULL XCL::ORIGINAL)
               (UNLESS XCL::NO-ERROR (FORMAT *ERROR-OUTPUT* "~S is not advised.~%" XCL::FN-TO-UNADVISE)
                )
              (NIL)
              (T (IL:PUTD XCL::EXECUTABLE-TO-UNADVISE (IL:GETD XCL::ORIGINAL)
                T)
                 (REMPROP XCL::EXECUTABLE-TO-UNADVISE 'IL:ADVISED)
                 (PUSH XCL::FN-TO-UNADVISE *UNADVISED-FNS*)
                 (SETQ IL:ADVISEDFNS (DELETE XCL::FN-TO-UNADVISE IL:ADVISEDFNS :TEST 'EQUAL))
                 (LIST XCL::FN-TO-UNADVISE))))
            (IF XCL::NO-IN-FN
                (ERROR "~S can't be selectively unadvised :IN ~S" XCL::FN-TO-UNADVISE XCL::IN-FN)
                (LET* ((XCL::EXECUTABLE-TO-UNADVISE-IN (XCL::NAME-OF-EXECUTABLE XCL::IN-FN))
                      (XCL::ADVICE-NAME ` (,XCL::FN-TO-UNADVISE :IN ,XCL::IN-FN))
                      (XCL::MIDDLE-MAN (GET-ADVICE-MIDDLE-MAN XCL::ADVICE-NAME)))
                    (COND

```

```

      ( (NULL XCL::MIDDLE-MAN)
        (UNLESS XCL::NO-ERROR (FORMAT *ERROR-OUTPUT* "~S is not advised.~%"
                                     XCL::ADVICE-NAME))
        NIL)
      (T (CHANGE-CALLS XCL::MIDDLE-MAN XCL::EXECUTABLE-TO-UNADVISE
                     XCL::EXECUTABLE-TO-UNADVISE-IN)
        (FINISH-UNADVISING XCL::ADVICE-NAME XCL::MIDDLE-MAN)
        (LIST XCL::ADVICE-NAME))))))

```

```

(DEFUN XCL:READVISE-FUNCTION (XCL::FN-TO-READVISE &KEY ((:IN XCL::IN-FN)))
  (MULTIPLE-VALUE-BIND (XCL::EXECUTABLE-TO-READVISE XCL::NO-IN-FN)
    (XCL::NAME-OF-EXECUTABLE XCL::FN-TO-READVISE)
    (COND
      ((AND (CONSP XCL::FN-TO-READVISE)
            (NOT XCL::EXECUTABLE-TO-READVISE))
        (IL:FOR XCL::FN IL:IN XCL::FN-TO-READVISE IL:JOIN (XCL:READVISE-FUNCTION XCL::FN :IN XCL::IN-FN)))
      ((AND (CONSP XCL::IN-FN)
            (NOT (XCL::NAME-OF-EXECUTABLE XCL::IN-FN)))
        (IL:FOR XCL::FN IL:IN XCL::IN-FN IL:JOIN (XCL:READVISE-FUNCTION XCL::FN-TO-READVISE :IN XCL::FN)))
      (T (XCL:UNADVISE-FUNCTION XCL::FN-TO-READVISE :IN XCL::IN-FN :NO-ERROR T)
        (IF XCL::IN-FN
          (FINISH-ADVISING XCL::FN-TO-READVISE XCL::EXECUTABLE-TO-READVISE XCL::IN-FN (
            XCL::NAME-OF-EXECUTABLE
            XCL::IN-FN)
          (FINISH-ADVISING XCL::FN-TO-READVISE XCL::EXECUTABLE-TO-READVISE))))))

```

```

(DEFUN UNADVISE-FROM-RESTORE-CALLS (FROM TO FN)
  (LET ((ENTRY (FIND-IF #'(LAMBDA (ENTRY)
                        (AND (CONSP ENTRY)
                            (EQ (FIRST ENTRY)
                                FROM)
                            (EQ (THIRD ENTRY)
                                FN))))
        IL:ADVISEDFNS)))
    (ASSERT (NOT (NULL ENTRY))
      NIL "BUG: Inconsistency in SI::UNADVISE-FROM-RESTORE-CALLS")
    (FINISH-UNADVISING ENTRY TO)
    (FORMAT *TERMINAL-IO* "~S unadvised.~%" ENTRY)))

```

```

(DEFUN FINISH-ADVISING (FN-TO-ADVISE EXECUTABLE-TO-ADVISE &OPTIONAL IN-FN EXECUTABLE-TO-ADVISE-IN)
  (COND
    ((NULL IN-FN)
      (LET* ((ALREADY-ADVISED? (MEMBER FN-TO-ADVISE IL:ADVISEDFNS :TEST 'EQ))
            (ORIGINAL (IF ALREADY-ADVISED?
                          (GET EXECUTABLE-TO-ADVISE-IN 'IL:ADVISED)
                          (LET ((*PRINT-CASE* :UPCASE))
                            (MAKE-SYMBOL (FORMAT NIL "Original ~A" EXECUTABLE-TO-ADVISE))))))
        ;; Adjust the database of advice for this function.
        (WHEN (NOT ALREADY-ADVISED?)
          (IL:PUTD ORIGINAL (IL:GETD EXECUTABLE-TO-ADVISE)
                    T))
          (IL:PUTD EXECUTABLE-TO-ADVISE (COMPILE NIL (CREATE-ADVISED-DEFINITION EXECUTABLE-TO-ADVISE ORIGINAL
                                                                                  FN-TO-ADVISE))))
        (WHEN (NOT ALREADY-ADVISED?)
          (SETF (GET EXECUTABLE-TO-ADVISE 'IL:ADVISED)
                ORIGINAL))
        ;; These are outside the WHEN because COMPILE calls VIRGINFN, which may unadvise the function.
        (SETQ *UNADVISED-FNS* (DELETE FN-TO-ADVISE *UNADVISED-FNS* :TEST 'EQUAL))
        (SETQ IL:ADVISEDFNS
              (CONS FN-TO-ADVISE (DELETE FN-TO-ADVISE IL:ADVISEDFNS :TEST 'EQUAL)))
          (IL:MARKASCHANGED FN-TO-ADVISE 'IL:ADVICE)
          (LIST FN-TO-ADVISE)))
      (T (LET* ((ADVICE-NAME `(,FN-TO-ADVISE :IN ,IN-FN))
              (ALREADY-ADVISED? (MEMBER ADVICE-NAME IL:ADVISEDFNS :TEST 'EQUAL))
              MIDDLE-MAN)
        ;; Create a middle-man for this request. If one has already been created, use it.
        (SETQ MIDDLE-MAN (OR (GET-ADVICE-MIDDLE-MAN ADVICE-NAME)
                            (SETF (GET-ADVICE-MIDDLE-MAN ADVICE-NAME)
                                  (CONSTRUCT-MIDDLE-MAN EXECUTABLE-TO-ADVISE EXECUTABLE-TO-ADVISE-IN))))
        ;; Give the middle-man the new advised definition.
        (IL:PUTD MIDDLE-MAN (COMPILE NIL (CREATE-ADVISED-DEFINITION EXECUTABLE-TO-ADVISE
                                                                      EXECUTABLE-TO-ADVISE ADVICE-NAME)))
        (WHEN (NOT ALREADY-ADVISED?)
          ;; Redirect any calls to FN-TO-ADVISE in IN-FN to call the middle-man.
          (CHANGE-CALLS EXECUTABLE-TO-ADVISE MIDDLE-MAN EXECUTABLE-TO-ADVISE-IN
                        'UNADVISE-FROM-RESTORE-CALLS))
        ;; Save a trail of information. These are outside the WHEN because COMPILE calls VIRGINFN, which may unadvise the function.

```





AROUND-FORMS :INITIAL-VALUE CALLING-FORM))

::  
;; Dealing with the File Manager

```
(IL:PUTDEF 'IL:ADVISE 'IL:FILEPKGCOMS
' ((IL:COM IL:MACRO (IL:X (IL:P IL:* (ADVISE-FILE-DEFINITIONS 'IL:X NIL)))
  IL:CONTENTS IL:NIL IL:ADD ADVISE-ADDTOCOM)
  (TYPE IL:DESCRIPTION "advice" IL:NEWCOM ADVISE-NEWCOM IL:GETDEF ADVISE-GETDEF IL:DELDEF ADVISE-DELDEF
  IL:PUTDEF ADVISE-PUTDEF IL:HASDEF ADVISE-HASDEF)))
```

```
(IL:PUTDEF 'IL:ADVISE 'IL:FILEPKGCOMS ' ((IL:COM IL:MACRO (IL:X (IL:P IL:* (ADVISE-FILE-DEFINITIONS 'IL:X T)))
  IL:CONTENTS ADVISE-CONTENTS IL:ADD ADVISE-ADDTOCOM)))
```

```
(DEFUN XCL:REINSTALL-ADVISE (XCL::NAME &KEY XCL::BEFORE XCL::AFTER XCL::AROUND)
  (IL:FOR XCL::ADVISE IL:IN XCL::BEFORE IL:DO (XCL:DESTRUCTURING-BIND (XCL::PRIORITY XCL::FORM)
    XCL::ADVISE
    (ADD-ADVISE XCL::NAME :BEFORE XCL::PRIORITY XCL::FORM)))
  (IL:FOR XCL::ADVISE IL:IN XCL::AFTER IL:DO (XCL:DESTRUCTURING-BIND (XCL::PRIORITY XCL::FORM)
    XCL::ADVISE
    (ADD-ADVISE XCL::NAME :AFTER XCL::PRIORITY XCL::FORM)))
  (IL:FOR XCL::ADVISE IL:IN XCL::AROUND IL:DO (XCL:DESTRUCTURING-BIND (XCL::PRIORITY XCL::FORM)
    XCL::ADVISE
    (ADD-ADVISE XCL::NAME :AROUND XCL::PRIORITY XCL::FORM))))
```

```
(DEFUN ADVISE-GETDEF (NAME TYPE OPTIONS)
  (LET ((ADVISE (CAR (GETHASH NAME *ADVISE-HASH-TABLE*)))
    (AND ADVISE (APPEND (IL:FOR ENTRY IL:IN (ADVISE-BEFORE ADVISE) IL:COLLECT (CONS ':BEFORE (COPY-TREE ENTRY)
    ))
    (IL:FOR ENTRY IL:IN (ADVISE-AFTER ADVISE) IL:COLLECT (CONS ':AFTER (COPY-TREE ENTRY)))
    (IL:FOR ENTRY IL:IN (ADVISE-AROUND ADVISE) IL:COLLECT (CONS ':AROUND (COPY-TREE ENTRY)
    ))))))
```

```
(DEFUN ADVISE-PUTDEF (NAME TYPE DEFINITION)
  (LET ((CANONICAL-DEFN (IL:FOR ENTRY IL:IN DEFINITION IL:COLLECT (LIST (CANONICALIZE-ADVISE-WHEN-SPEC
    (CAR ENTRY))
    (CANONICALIZE-ADVISE-WHERE-SPEC
    (COPY-TREE (CADR ENTRY)))
    (COPY-TREE (CADDR ENTRY))))))
    (CURRENT-ADVISE (OR (CAR (GETHASH NAME *ADVISE-HASH-TABLE*))
    (CAR (SETF (GETHASH NAME *ADVISE-HASH-TABLE*)
    (CONS (MAKE-ADVISE)
    NIL))))))
    (SETF (ADVISE-BEFORE CURRENT-ADVISE)
    (MAPCAR #'REST (IL:FOR ENTRY IL:IN CANONICAL-DEFN IL:WHEN (EQ (CAR ENTRY)
    :BEFORE)
    IL:COLLECT ENTRY)))
    (SETF (ADVISE-AFTER CURRENT-ADVISE)
    (MAPCAR #'REST (IL:FOR ENTRY IL:IN CANONICAL-DEFN IL:WHEN (EQ (CAR ENTRY)
    :AFTER)
    IL:COLLECT ENTRY)))
    (SETF (ADVISE-AROUND CURRENT-ADVISE)
    (MAPCAR #'REST (IL:FOR ENTRY IL:IN CANONICAL-DEFN IL:WHEN (EQ (CAR ENTRY)
    :AROUND)
    IL:COLLECT ENTRY)))
    (IF (CONSP NAME)
    (XCL:READADVISE-FUNCTION (FIRST NAME)
    :IN
    (THIRD NAME))
    (XCL:READADVISE-FUNCTION NAME))))
```

```
(DEFUN ADVISE-DELDEF (NAME TYPE)
  (DECLARE (IGNORE TYPE))
  (WHEN (MEMBER NAME IL:ADVISED FNS :TEST 'EQUAL)
  (IF (CONSP NAME)
  (XCL:UNADVISE-FUNCTION (FIRST NAME)
  :IN
  (THIRD NAME))
  (XCL:UNADVISE-FUNCTION NAME))
  (FORMAT *TERMINAL-IO* "~S unadvised." NAME))
  (REMHASH NAME *ADVISE-HASH-TABLE*))
```

```
(DEFUN ADVISE-HASDEF (NAME TYPE SOURCE)
  (AND (GETHASH NAME *ADVISE-HASH-TABLE*)
  (OR NAME T)))
```

```
(DEFUN ADVISE-NEWCOM (NAME TYPE LISTNAME FILE)
```

;;; If you make a new com for ADVISE, you should make an ADVISE command.

```
(IL:DEFAULTMAKENEWCOM NAME 'IL:ADVISE LISTNAME FILE))
```

```
(DEFUN ADVICE-FILE-DEFINITIONS (NAMES READWISE?)
```

;;; READWISE? is true for the File Manager command ADVISE and false for the command ADVICE. For ADVISE, we want to emit a form to readwise the  
;;; named functions after reinstalling the advice.

```
(LET ((REAL-NAMES NIL))
  `,@(IL:FOR FN IL:IN NAMES
    IL:COLLECT (LET* ((NAME (IL:IF (CONSP FN)
                                  IL:THEN (ASSERT (AND (EQ (SECOND FN)
                                                         :IN)
                                                         (= 3 (LENGTH FN)))
                                                         NIL "~S should be of the form (FOO :IN BAR)" FN)
                                  FN)
                    IL:ELSE (LET ((NAME (CANONICALIZE-ADVICE-SYMBOL FN))
                                (OLD-ADVICE (GET FN 'IL:READWISE)))
                              (WHEN OLD-ADVICE
                                (ADD-OLD-STYLE-ADVICE NAME OLD-ADVICE)
                                (REMPROP FN 'IL:READWISE))
                              NAME)))
                  (ADVISE (CAR (GETHASH NAME *ADVISE-HASH-TABLE*))))
                (ASSERT (NOT (NULL ADVICE))
                  NIL "Can't find advice for ~S" NAME)
                (PUSH NAME REAL-NAMES)
                `(XCL:REINSTALL-ADVICE ',NAME
                  ,@(AND (ADVISE-BEFORE ADVICE)
                        `(:BEFORE ', (ADVISE-BEFORE ADVICE)))
                  ,@(AND (ADVISE-AFTER ADVICE)
                        `(:AFTER ', (ADVISE-AFTER ADVICE)))
                  ,@(AND (ADVISE-AROUND ADVICE)
                        `(:AROUND ', (ADVISE-AROUND ADVICE))))))
    ,@(AND READWISE? `(IL:READWISE ,@(REVERSE REAL-NAMES))))))
```

```
(DEFUN ADVICE-CONTENTS (COM NAME TYPE)
```

```
(AND (EQ TYPE 'IL:ADVISE)
  (COND
    ((NULL NAME) ; Return a list of the ADVICE's in the given COM.
     (CDR COM))
    ((EQ NAME 'T) ; Return T if there are ANY ADVICE's in the given COM.
     (NOT (NULL (CDR COM))))
    ((OR (SYMBOLP NAME)
         (= (LENGTH NAME)
            3)
         (EQ (SECOND NAME)
              :IN)) ; Return T iff an ADVICE named NAME in the given COM.
     (AND (MEMBER NAME (CDR COM)
                   :TEST
                   'EQUAL)
          T))
    (T) ; NAME is a list of names. Return the intersection of that list with
         ; the ADVICE's in the given COM.
     (INTERSECTION NAME (CDR COM)
                   :TEST
                   'EQUAL))))
```

```
(DEFUN ADVICE-ADDTOCOM (COM NAME TYPE NEAR)
```

;;; This is the ADD method for both of the ADVICE and ADVISE commands.

;;; Add the given name only if the type is ADVISE. Also, add it to ADVICE commands only if a NEAR was specified. We want to normally create only  
;;; ADVISE commands. If the user really wants an ADVICE command, they'll have to create it themselves.

```
(AND (EQ TYPE 'IL:ADVISE)
  (OR (EQ (CAR COM)
         'IL:ADVISE)
      (NOT (NULL NEAR))))
(IL:ADDTOCOM1 COM NAME NEAR NIL))
```

```
(IL:PUTPROPS IL:ADVISED IL:PROPTYPE IGNORE)
```

;;  
;; Dealing with old-style advice

```
(DEFUN IL:READWISE1 (IL:FN)
```

```
(FLET ((IL:READWISE-ENTRY (IL:ENTRY)
  (IL:IF (IL:LISTP IL:ENTRY)
    IL:THEN (XCL:READWISE-FUNCTION (FIRST IL:ENTRY)
                                   :IN
                                   (THIRD IL:ENTRY))
    IL:ELSE (XCL:READWISE-FUNCTION IL:ENTRY)))
  (IL:IF (IL:LISTP IL:FN)
```



```

IL:THEN (ASSERT (IL:STRING.EQUAL (SECOND IL:FN)
                                "IN")
          (IL:READVICE-ENTRY IL:FN)
          NIL "~S should be in the form (FOO IN BAR).~%" IL:FN)
IL:ELSE (LET ((IL:NAME (CANONICALIZE-ADVICE-SYMBOL IL:FN))
              (IL:OLD-ADVICE (GET IL:FN 'IL:READVICE)))
        (IL:IF IL:OLD-ADVICE
              IL:THEN (ADD-OLD-STYLE-ADVICE IL:NAME IL:OLD-ADVICE)
              (REMPROP IL:FN 'IL:READVICE))
        (IL:READVICE-ENTRY IL:NAME))))

```

```

(DEFUN ADD-OLD-STYLE-ADVICE (NAME OLD-ADVICE)

```

;; OLD-ADVICE should be the value of the READVICE property of some symbol. Note that the CAR of that value is the old middle-man used for -IN-  
 ;; advice. Thus, we take the CDR below.

```

(WHEN (NOT (MEMBER NAME IL:ADVISED-FNS :TEST 'EQUAL))
      (DELETE-ADVICE NAME))
(IL:FOR ADVICE IL:IN (CDR OLD-ADVICE) IL:DO (XCL:DESTRUCTURING-BIND (WHEN WHERE WHAT)
                                                                    ADVICE
                                                                    ;; Translate Interlisp names to the new standard.
                                                                    (ADD-ADVICE NAME (CANONICALIZE-ADVICE-WHEN-SPEC WHEN)
                                                                    (CANONICALIZE-ADVICE-WHERE-SPEC WHERE)
                                                                    WHAT))))

```

```

(DEFUN CANONICALIZE-ADVICE-SYMBOL (SYMBOL)
  (LET ((IN-POS (IL:STRPOS "-IN-" SYMBOL)))
    (IF (NULL IN-POS)
        SYMBOL
        (LIST (IL:SUBATOM SYMBOL 1 (1- IN-POS))
              :IN
              (IL:SUBATOM SYMBOL (+ IN-POS 4)
                              NIL))))))

```

```

(DEFUN CANONICALIZE-ADVICE-WHEN-SPEC (SPEC)
  (IF (NULL SPEC)
      ' :BEFORE
      (INTERN (STRING SPEC)
              "KEYWORD")))

```

```

(DEFUN CANONICALIZE-ADVICE-WHERE-SPEC (SPEC)
  (CASE SPEC
    ((NIL LAST IL:BOTTOM IL:END :LAST) ' :LAST)
    ((IL:TOP IL:FIRST :FIRST) ' :FIRST)
    (T (IF (CONSP SPEC)
           SPEC
           (ERROR "Illegal WHERE specification to ADVISE: ~S" SPEC)))))

```

```

(XCL:DEF-DEFINE-TYPE XCL:ADVISED-FUNCTIONS "Advised function definitions")

```

```

(XCL:DEFDEFINER (XCL:DEFADVICE (:PROTOTYPE (LAMBDA (XCL::NAME)
                                             \ (XCL:DEFADVICE ,XCL::NAME
                                             "advice"))))
  XCL:ADVISED-FUNCTIONS (XCL::NAME &BODY XCL::ADVICE-FORMS)
  \ (PROGN ,. (XCL:WITH-COLLECTION
              (DOLIST (XCL::ADVICE XCL::ADVICE-FORMS)
                    (XCL:COLLECT (XCL:DESTRUCTURING-BIND
                                  (XCL::FN-TO-ADVISE XCL::FORM &KEY XCL::IN WHEN XCL::PRIORITY)
                                  XCL::ADVICE
                                  \ (XCL:ADVISE-FUNCTION ',XCL::FN-TO-ADVISE ',XCL::FORM
                                  ,@ (AND XCL::IN `(:IN ',XCL::IN))
                                  ,@ (AND WHEN `(:WHEN ,WHEN))
                                  ,@ (AND XCL::PRIORITY `(:PRIORITY ,XCL::PRIORITY))))))))))

```

;; Arrange for the proper package. Because of the DEFSTRUCT above, we must have the file dumped in the SYSTEM package.

```

(IL:PUTPROPS IL:ADVISE IL:MAKEFILE-ENVIRONMENT (:READTABLE "XCL" :PACKAGE "SYSTEM"))
(IL:PUTPROPS IL:ADVISE IL:FILETYPE :COMPILE-FILE)
(IL:DECLARE\ IL:DONTEVAL@LOAD IL:DOEVAL@COMPILE IL:DONTCOPY IL:COMPILERVERS
(IL:ADDTOVAR IL:NLAMA IL:READVICE IL:UNADVISE)
(IL:ADDTOVAR IL:NLAML )
(IL:ADDTOVAR IL:LAMA IL:ADVISE)
)
(IL:RPAQQ IL:ADVISECOMS
  ((IL:STRUCTURES ADVICE)

```

```

(IL:VARIABLES IL:ADVISED-FNS *UNADVISED-FNS*)
;;
;; Interlisp entry points.
(IL:FNS IL:ADVISE IL:UNADVISE IL:READADVISE)
(IL:PROP IL:ARGNAMES IL:ADVISE)
;;
;; XCL entry points.
(IL:FUNCTIONS XCL:ADVISE-FUNCTION XCL:UNADVISE-FUNCTION XCL:READADVISE-FUNCTION)
(IL:FUNCTIONS UNADVISE-FROM-RESTORE-CALLS FINISH-ADVISING FINISH-UNADVISING)
;;
;; The advice database.
(IL:VARIABLES *ADVICE-HASH-TABLE*)
(IL:FUNCTIONS ADD-ADVICE DELETE-ADVICE GET-ADVICE-MIDDLE-MAN SET-ADVICE-MIDDLE-MAN INSERT-ADVICE-FORM
)
(IL:SETFS GET-ADVICE-MIDDLE-MAN)
;;
;; Hacking the actual advice forms.
(IL:FUNCTIONS CREATE-ADVISED-DEFINITION MAKE-AROUND-BODY)
;;
;; Dealing with the File Manager
(IL:FILEPKGCOMS IL:ADVISE IL:ADVISE)
(IL:FUNCTIONS XCL:REINSTALL-ADVICE)
(IL:FUNCTIONS ADVISE-GETDEF ADVISE-PUTDEF ADVISE-DELDEF ADVISE-HASDEF ADVISE-NEWCOM
ADVISE-FILE-DEFINITIONS ADVISE-CONTENTS ADVISE-ADDTOCOM)
(IL:PROP IL:PROPTYPE IL:ADVISED)
;;
;; Dealing with old-style advice
(IL:FUNCTIONS IL:READADVISE1 ADD-OLD-STYLE-ADVICE CANONICALIZE-ADVICE-SYMBOL
CANONICALIZE-ADVICE-WHEN-SPEC CANONICALIZE-ADVICE-WHERE-SPEC)
(IL:DEFINE-TYPES XCL:ADVISED-FUNCTIONS)
(IL:FUNCTIONS XCL:DEFADVISE)
;; Arrange for the proper package. Because of the DEFSTRUCT above, we must have the file dumped in the SYSTEM package.
(IL:PROP (IL:MAKEFILE-ENVIRONMENT IL:FILETYPE)
IL:ADVISE)
(IL:DECLARE\ : IL:DONTEVAL@LOAD IL:DOEVAL@COMPILE IL:DONTCOPY IL:COMPILEVAR (IL:ADDVARS (IL:NLAMA)
(IL:NLAML)
(IL:LAMA
IL:ADVISE)
))))
(IL:DECLARE\ : IL:DONTEVAL@LOAD IL:DOEVAL@COMPILE IL:DONTCOPY IL:COMPILEVAR
(IL:ADDTOVAR IL:NLAMA )
(IL:ADDTOVAR IL:NLAML )
(IL:ADDTOVAR IL:LAMA IL:ADVISE)
)
(IL:PUTPROPS IL:ADVISE IL:COPYRIGHT ("Venue & Xerox Corporation" T 1978 1984 1986 1987 1988 1990 1992))

```

---

**FUNCTION INDEX**

ADD-ADVICE .....	5	ADVISE-CONTENTS .....	8	INSERT-ADVICE-FORM .....	5
ADD-OLD-STYLE-ADVICE .....	9	XCL:ADVISE-FUNCTION .....	2	MAKE-AROUND-BODY .....	6
ADVICE-ADDTOCOM .....	8	CANONICALIZE-ADVICE-SYMBOL .....	9	IL:READVICE .....	2
ADVICE-DELDEF .....	7	CANONICALIZE-ADVICE-WHEN-SPEC .....	9	XCL:READVICE-FUNCTION .....	4
ADVICE-FILE-DEFINITIONS .....	8	CANONICALIZE-ADVICE-WHERE-SPEC .....	9	IL:READVICE1 .....	8
ADVICE-GETDEF .....	7	CREATE-ADVISED-DEFINITION .....	6	XCL:REINSTALL-ADVICE .....	7
ADVICE-HASDEF .....	7	DELETE-ADVICE .....	5	SET-ADVICE-MIDDLE-MAN .....	5
ADVICE-NEWCOM .....	7	FINISH-ADVISING .....	4	IL:UNADVISE .....	2
ADVICE-PUTDEF .....	7	FINISH-UNADVISING .....	5	UNADVISE-FROM-RESTORE-CALLS .....	4
IL:ADVISE .....	2	GET-ADVICE-MIDDLE-MAN .....	5	XCL:UNADVISE-FUNCTION .....	3

---

**VARIABLE INDEX**

*ADVISE-HASH-TABLE* .....	5	*UNADVISED-FNS* .....	1	IL:ADVISED-FNS .....	1
---------------------------	---	-----------------------	---	----------------------	---

---

**PROPERTY INDEX**

IL:ADVISE .....	2,9	IL:ADVISED .....	8
-----------------	-----	------------------	---

---

**DEFINER INDEX**

XCL:DEFADVISE .....	9
---------------------	---

---

**DEFINE-TYPE INDEX**

XCL:ADVISED-FUNCTIONS .....	9
-----------------------------	---

---

**SETF INDEX**

GET-ADVICE-MIDDLE-MAN .....	6
-----------------------------	---

---

**STRUCTURE INDEX**

ADVISE .....	1
--------------	---

---