File created: 23-Dec-87 16:56:46 **{FireFS:CS:Univ Rochester}<Koomen>LispUsers>Lyric>PORT-CLFILE.;14**

changes to: (IL:FUNCTIONS EXPORT-CLFILE)

previous date: 3-Nov-87 16:27:35 {FireFS:CS:Univ Rochester}<Koomen>LispUsers>Lyric>PORT-CLFILE.;12

Read Table: XCL

Package: XEROX-COMMON-LISP

Format: XCCS

; Copyright (c) 1987 by Johannes Koomen, Larry Masinter. All rights reserved.

```
(IL:RPAQQ IL:PORT-CLFILECOMS
          ((IL:FUNCTIONS IMPORT-CLFILE EXPORT-CLFILE)
           (IL:P (EXPORT '(IMPORT-CLFILE EXPORT-CLFILE)
                         (FIND-PACKAGE "XCL"))
                 (IMPORT '(IMPORT-CLFILE EXPORT-CLFILE)
                         (FIND-PACKAGE "INTERLISP")))
           (IL:FUNCTIONS CLFILE-EXPORT-FILECOM CLFILE-PARSE-FORM CLFILE-PARSE-MODE CLFILE-READ-SEMI
                  CLFILE-SET-MODE)
           (IL:PROP (IL:FILETYPE IL:MAKEFILE-ENVIRONMENT)
                  IL:PORT-CLFILE)))


(DEFUN IMPORT-CLFILE (FILEPATH)
   "Load a standard CommonLisp file FILEPATH, creating COMS for FILEPATH"
   (WITH-OPEN-FILE (*STANDARD-INPUT* FILEPATH :DIRECTION :INPUT)
          (DECLARE (SPECIAL *STANDARD-INPUT*))
          (LET ((*PACKAGE* (FIND-PACKAGE "USER"))
                (*READTABLE* (COPY-READTABLE NIL))
                (*READ-BASE* 10)
                (COMSLST NIL)
                (EOF-VALUE (LIST :EOF))
                (FILE-ID (INTERN (STRING-UPCASE (PATHNAME-NAME *STANDARD-INPUT*))
                                 (FIND-PACKAGE "INTERLISP"))))
               (DECLARE (SPECIAL *PACKAGE* *READTABLE* *READ-BASE*))
               ;; Copy readtable, change ; macro to preserve comments
               (SET-MACRO-CHARACTER #\; #'CLFILE-READ-SEMI NIL *READTABLE*)
               (DO ((CLFORM (READ NIL NIL EOF-VALUE)
                            (READ NIL NIL EOF-VALUE))
                    (FIRSTCLFORMP T)
                    (NEXTCOMSENTRY NIL)
                    (LASTCOMSENTRY NIL)
                    (NEXTCOMSWEIRDP NIL))
                   ((EQ CLFORM EOF-VALUE)
                    (IF LASTCOMSENTRY (PUSH LASTCOMSENTRY COMSLST))
                    (SETQ COMSLST (NREVERSE COMSLST)))
                 (WHEN FIRSTCLFORMP
                     (SETQ FIRSTCLFORMP NIL)
                     (COND
                        ((AND (CONSP CLFORM)
                              (EQ (CAR CLFORM)
                                  'IL:*)
                              (DO ((CLTAIL (CDR CLFORM)
                                           (CDR CLTAIL)))
                                  ((NOT (CONSP CLTAIL))
                                   NIL)
                                 (IF (STRINGP (CAR CLTAIL))
                                     (RETURN (CLFILE-PARSE-MODE (CAR CLTAIL)
                                                                FILE-ID)))))
                         (SETQ CLFORM NIL))))
                 (UNLESS (NULL CLFORM)
                     (EVAL CLFORM)
                     (SETQ NEXTCOMSENTRY (CLFILE-PARSE-FORM CLFORM))
                     (SETQ NEXTCOMSWEIRDP (OR (EQ (CAR NEXTCOMSENTRY)
                                                  'IL:*)
                                              (CONSP (CDDR NEXTCOMSENTRY))))
                     (WHEN (AND LASTCOMSENTRY (OR NEXTCOMSWEIRDP (NOT (EQ (CAR NEXTCOMSENTRY)
                                                                           (CAR LASTCOMSENTRY)))))
                         (PUSH LASTCOMSENTRY COMSLST)
                         (SETQ LASTCOMSENTRY NIL))
                     (COND
                        (NEXTCOMSWEIRDP (PUSH NEXTCOMSENTRY COMSLST))
                        (LASTCOMSENTRY (NCONC LASTCOMSENTRY (CDR NEXTCOMSENTRY)))
                        (T (SETQ LASTCOMSENTRY NEXTCOMSENTRY)))))
               (SETF (GET FILE-ID 'IL:FILETYPE)
                     :COMPILE-FILE)
               (SET (IL:FILECOMS FILE-ID)
                    (NCONC COMSLST `((IL:PROP (IL:FILETYPE IL:MAKEFILE-ENVIRONMENT)
                                              ,FILE-ID))))
               (PUSHNEW FILE-ID IL:FILELST)
               (IL:MARKASCHANGED FILE-ID 'IL:FILES 'IL:DEFINED)
               FILE-ID)))
```

```
(DEFUN EXPORT-CLFILE (FILEPATH &OPTIONAL (LINELENGTH 72))
   "Write a standard CommonLisp file FILEPATH, using COMS for FILEPATH"
   (DECLARE (GLOBAL IL:FILELINELENGTH))
   (IL:RESETVARS ((IL:FILELINELENGTH LINELENGTH))
                 (RETURN (WITH-OPEN-FILE (*STANDARD-OUTPUT* (MAKE-PATHNAME :TYPE "LISP" :VERSION :NEWEST
                                                                           :DEFAULTS FILEPATH)
                                                            :DIRECTION :OUTPUT)
                                         (DECLARE (SPECIAL *STANDARD-OUTPUT*))
                                         (LET ((*PACKAGE* (FIND-PACKAGE "USER"))
                                               (*READTABLE* (IL:FIND-READTABLE "LISP"))
                                               (*PRINT-BASE* 10)
                                               (*PRINT-ARRAY* T)
                                               (*PRINT-LEVEL* NIL)
                                               (*PRINT-LENGTH* NIL)
                                               (IL:FONTCHANGEFLG NIL)
                                               (IL:\#RPARS NIL)
                                               (IL:**COMMENT**FLG NIL)
                                               (IL:*PRINT-SEMICOLON-COMMENTS* T)
                                               (IL:*PRINT-STRUCTURE* T)
                                               (FILE-ID (INTERN (STRING-UPCASE (PATHNAME-NAME *STANDARD-OUTPUT*))
                                                                (FIND-PACKAGE "INTERLISP"))))
                                              (DECLARE (SPECIAL *PACKAGE* *READTABLE* *PRINT-BASE* *PRINT-ARRAY*
                                                               *PRINT-LEVEL* *PRINT-LENGTH* IL:FONTCHANGEFLG IL:\#RPARS
                                                               IL:**COMMENT**FLG IL:*PRINT-SEMICOLON-COMMENTS*
                                                               IL:*PRINT-STRUCTURE*))
                                              (CLFILE-SET-MODE (GET FILE-ID 'IL:MAKEFILE-ENVIRONMENT))
                                              (FORMAT T ";;; -*- Package: ~A; Syntax: ~A; Mode: Lisp; Base: ~D -*-"
                                                     (STRING-CAPITALIZE (PACKAGE-NAME *PACKAGE*))
                                                     (LET ((RDTBLNAME (IL:READTABLEPROP *READTABLE* 'IL:NAME)))
                                                          (COND
                                                             ((OR (NOT (STRINGP RDTBLNAME))
                                                                  (STRING-EQUAL RDTBLNAME "XCL")
                                                                  (STRING-EQUAL RDTBLNAME "LISP"))
                                                              "Common-Lisp")
                                                             (T (STRING-CAPITALIZE RDTBLNAME))))
                                                     *PRINT-BASE*)
                                              (FORMAT T "~2%;;; File converted on ~A from source ~A" (IL:DATE
                                                                                                      (IL:DATEFORMAT

                                                                                                     IL:NO.LEADING.SPACES
                                                                                                      ))
                                                     (SYMBOL-NAME FILE-ID))
                                              (LET ((DATES (GET FILE-ID 'IL:FILEDATES)))
                                                   (WHEN DATES
                                                      (FORMAT T "~&;;; Original source ~A created ~A" (CDAR DATES)
                                                              (CAAR DATES))))
                                              (TERPRI)
                                              (TERPRI)
                                              (IL:PRINTCOPYRIGHT FILE-ID)
                                              (MAPC #'CLFILE-EXPORT-FILECOM (IL:LISTP (IL:GETTOPVAL (IL:FILECOMS FILE-ID)
                                                                                                    )))
                                              (NAMESTRING *STANDARD-OUTPUT*))))))

(EXPORT '(IMPORT-CLFILE EXPORT-CLFILE)
       (FIND-PACKAGE "XCL"))

(IMPORT '(IMPORT-CLFILE EXPORT-CLFILE)
       (FIND-PACKAGE "INTERLISP"))


(DEFUN CLFILE-EXPORT-FILECOM (COMMAND)
   (FLET ((SAVE-PROP (SYMBOL PROP VAL)
                (CASE PROP
                    ((IL:FILETYPE IL:MAKEFILE-ENVIRONMENT)      ;IGNORE
                       NIL)
                    (T (PPRINT `(SETF (GET ',SYMBOL ',PROP)
                                      ',VAL))))))
          (CASE (IL:GETFILEPKGTYPE (CAR COMMAND)
                       'COMMAND)
              (IL:FNS (MAPC #'(LAMBDA (FN)
                                      (PPRINT (LET ((DEF (IL:GETDEF FN 'IL:FNS)))
                                                   (ECASE (CAR DEF)
                                                       ((LAMBDA) `(DEFUN ,FN ,@(CDR DEF)))
                                                       ((IL:LAMBDA) `(DEFUN ,FN (&OPTIONAL ,@(SECOND DEF))
                                                                            ,@(CDDR DEF)))))))
                            (IL:PRETTYCOM1 COMMAND T T)))
              (IL:VARS (MAPC #'(LAMBDA (VAR)
                                       (IF (LISTP VAR)
                                           `(DEFPARAMETER (FIRST VAR) (SECOND VAR))
                                           `(DEFPARAMETER ,VAR ',(IL:GETTOPVAL VAR))))
                             (IL:PRETTYCOM1 COMMAND T T)))
              (IL:DECLARE\: (LET ((CONTEXT '(LOAD EVAL))
                                  (WHEN-CLAUSE T))
                                 (DO ((TAIL (IL:PRETTYCOM1 COMMAND T T)
                                            (CDR TAIL)))
                                     ((NULL TAIL))
                                   (CASE (CAR TAIL)
```

```
                                ((IL:EVAL@LOADWHEN)
                                   (PUSHNEW 'EVAL CONTEXT)
                                   (SETQ WHEN-CLAUSE (IF (EQ WHEN-CLAUSE T)
                                                         (CADR TAIL)
                                                         `(AND ,(CADR TAIL)
                                                               ,WHEN-CLAUSE)))
                                   (SETQ TAIL (CDR TAIL)) ; consumes two tokens, one by the DO

                                   )
                                ((IL:EVAL@COMPILEWHEN)
                                   (PUSHNEW 'COMPILE CONTEXT)
                                   (SETQ WHEN-CLAUSE (IF (EQ WHEN-CLAUSE T)
                                                         (CADR TAIL)
                                                         `(AND ,(CADR TAIL)
                                                               ,WHEN-CLAUSE)))
                                   (SETQ TAIL (CDR TAIL)))
                                ((IL:COPYWHEN)
                                   (PUSHNEW 'LOAD CONTEXT)
                                   (SETQ WHEN-CLAUSE (IF (EQ WHEN-CLAUSE T)
                                                         (CADR TAIL)
                                                         `(AND ,(CADR TAIL)
                                                               ,WHEN-CLAUSE)))
                                   (SETQ TAIL (CDR TAIL)))
                                ((IL:FIRST IL:NOTFIRST)   ; IGNORE
                                                          )
                                ((IL:COMPILERVARS)
                                   ;; throw these out
                                   (RETURN-FROM CLFILE-EXPORT-FILECOM NIL))
                                ((IL:COPY IL:DOCOPY) (PUSHNEW 'LOAD CONTEXT))
                                ((IL:DOEVAL@COMPILE IL:EVAL@COMPILE) (PUSHNEW 'COMPILE CONTEXT))
                                ((IL:DOEVAL@LOAD IL:EVAL@LOAD) (PUSHNEW 'EVAL CONTEXT))
                                ((IL:DONTCOPY) (SETQ CONTEXT (REMOVE 'LOAD CONTEXT)))
                                ((IL:DONTEVAL@COMPILE) (SETQ CONTEXT (REMOVE 'COMPILE CONTEXT)))
                                ((IL:DONTEVAL@LOAD) (SETQ CONTEXT (REMOVE 'EVAL CONTEXT)))
                                (T (FORMAT T "~&(eval-when &S " CONTEXT)
                                   (CLFILE-EXPORT-FILECOM (CAR TAIL))
                                   (FORMAT T ")")))))))
((IL:SPECVARS) (PPRINT `(PROCLAIM '(SPECIAL ,@(IL:PRETTYCOM1 COMMAND T T)))))
((IL:GLOBALVARS) (PPRINT `(PROCLAIM '(GLOBAL ,@(IL:PRETTYCOM1 COMMAND T T)))))
((IL:LOCALVARS) (PPRINT `(PROCLAIM '(LEXICAL ,@(IL:PRETTYCOM1 COMMAND T T)))))
((IL:PROP IL:IFPROP) (PROG ((OPTIONAL (EQ (CAR COMMAND)
                                          'IL:IFPROP))
                            (PROPS (CADR COMMAND))
                            (NOT-FOUND "NOT ON ANY PROPERTY LIST")
                            (SYMBOLS (IL:PRETTYCOM1 (CDR COMMAND)
                                                    T T))) ; IFPROP only dumps those property values that are non-NIL.
                           (MAPC #'(LAMBDA (SYMBOL)
                                           (DECLARE (SPECIAL IL:SYSPROPS))
                                           (FLET ((DO-PROP (PROP)
                                                           (UNLESS
                                                               (AND OPTIONAL
                                                                    (EQ NOT-FOUND
                                                                        (GET SYMBOL PROP NOT-FOUND)))
                                                               (SAVE-PROP SYMBOL PROP (GET SYMBOL PROP
                                                                                           )))))
                                                  (COND
                                                     ((CONSP PROPS)
                                                      (MAPC #'DO-PROP PROPS))
                                                     ((EQ PROPS 'IL:ALL)
                                                      (DO ((TAIL (SYMBOL-PLIST SYMBOL)
                                                                 (CDDR TAIL)))
                                                          ((NULL TAIL))
                                                         (UNLESS (MEMBER (CAR TAIL)
                                                                         IL:SYSPROPS)
                                                                 (DO-PROP (CAR TAIL)))))
                                                     (T (DO-PROP PROPS))))))
                                 SYMBOLS)))
(IL:P (MAPC #'(LAMBDA (X)
                      (CASE (CAR X)
                          ((IL:PUTPROPS) (DO ((TAIL (CDR X)
                                                    (CDDDR TAIL)))
                                             ((NULL TAIL))
                                           (SAVE-PROP (FIRST TAIL)
                                                  (SECOND TAIL)
                                                  (THIRD TAIL))))
                          (T (PPRINT X))))
            (IL:PRETTYCOM1 COMMAND T)))
(IL:INITVARS (MAPC #'(LAMBDA (X)
                             (DECLARE (SPECIAL IL:COMMENTFLG))
                             (PPRINT (COND
                                        ((LISTP X)
                                         (IF (EQ (CAR X)
                                                 IL:COMMENTFLG)
                                             X
                                             `(DEFVAR ,@X)))
                                        (T (HELP)))))
```

```
                              (IL:PRETTYCOM1 COMMAND T T)))
            (IL:COMS (MAPC #'CLFILE-EXPORT-FILECOM (IL:PRETTYCOM1 COMMAND T)))
            ((IL:*)
               (COND
                  ((EQ (CADR COMMAND)
                       'IL:*)                                    ; Form-feed if super-comment indicated.  Use * no matter what
                                                                 ; current COMMENTFLG is.
                   (WRITE-CHAR #\Page)))
               (PPRINT COMMAND))
            (T (LET ((DEF (CDR (ASSOC (CAR COMMAND)
                                      IL:PRETTYDEFMACROS))))
                    (IF DEF
                        (MAPC #'CLFILE-EXPORT-FILECOM (IL:SUBPAIR (CAR DEF)
                                                                  (IL:PRETTYCOM1 COMMAND T T)
                                                                  (CDR DEF)))
                        (HELP "CAN'T HANDLE" (CAR COMMAND))))))))


(DEFUN CLFILE-PARSE-FORM (CLFORM)
   "Given CommonLisp FORM, creates (filepkgtype object)"
   (COND
      ((OR (NOT (CONSP CLFORM))
           (NOT (SYMBOLP (CAR CLFORM))))
       `(IL:P ,CLFORM))
      ((EQ (CAR CLFORM)
           'IL:*)
       CLFORM)
      ((LET ((COMSTYPE (GET (CAR CLFORM)
                            'IL:DEFINER-FOR))
             (COMSNAME (CADR CLFORM)))
            (IF COMSTYPE
                (LIST COMSTYPE (IF (CONSP COMSNAME)              ; e.g. STRUCTURES
                                   (CAR COMSNAME)
                                   COMSNAME)))))
      ((CASE (CAR CLFORM)
           (PROCLAIM (LET ((DECLSPEC (CAR (IL:CONSTANTEXPRESSIONP (CADR CLFORM)))))
                          (CASE (CAR DECLSPEC)
                              (GLOBAL (CONS 'IL:GLOBALVARS (CDR DECLSPEC)))
                              (SPECIAL (CONS 'IL:SPECVARS (CDR DECLSPEC)))
                              (LEXICAL (CONS 'IL:LOCALVARS (CDR DECLSPEC))))))
           (EVAL-WHEN (LET (EVALFLG LOADFLG COMPILEFLG)
                           (DO ((CONTEXT (CADR CLFORM)
                                         (CDR CONTEXT)))
                               ((NULL CONTEXT))
                             (CASE (CAR CONTEXT)
                                 (EVAL (SETQ EVALFLG T))
                                 (LOAD (SETQ LOADFLG T))
                                 (COMPILE (SETQ COMPILEFLG T))))
                           `(IL:DECLARE\: ,(IF EVALFLG
                                               'IL:EVAL@LOAD
                                               'IL:DONTEVAL@LOAD)
                                          ,(IF LOADFLG
                                               'IL:COPY
                                               'IL:DONTCOPY)
                                          ,(IF COMPILEFLG
                                               'IL:EVAL@COMPILE
                                               'IL:DONTEVAL@COMPILE)
                                          (COMS ,@(MAPCAR #'CLFILE-PARSE-FORM (CADDR CLFORM))))))))
      (T `(IL:P ,CLFORM))))


(DEFUN CLFILE-PARSE-MODE (MODE-STRING &OPTIONAL FILE-ID)
   "Setf *PACKAGE*, *READTABLE* and *READ-BASE* according to file mode comment"
   (DECLARE (SPECIAL *PACKAGE* *READTABLE* *READ-BASE*))
   (WHEN (SEARCH "-*-" MODE-STRING :END2 3)
       (PROG ((MODESTR (STRING-UPCASE MODE-STRING))
              MODEPOS MODENAME OBJECT)
             (WHEN (SETQ MODEPOS (SEARCH "PACKAGE: " MODESTR))
                 (SETQ MODENAME (STRING (READ-FROM-STRING MODESTR NIL NIL :START (+ MODEPOS (LENGTH "PACKAGE: ")
                                                                                    ))))
                 (SETQ OBJECT (FIND-PACKAGE MODENAME))
                 (COND
                    ((PACKAGEP OBJECT)
                     (SETQ *PACKAGE* OBJECT))
                    (T (ERROR "~&Non-existent package: ~A~%" MODENAME))))
             (WHEN (SETQ MODEPOS (SEARCH "SYNTAX: " MODESTR))
                 (SETQ MODENAME (STRING (READ-FROM-STRING MODESTR NIL NIL :START (+ MODEPOS (LENGTH "SYNTAX: "))
                                                         )))
                 (IF (STRING-EQUAL MODENAME "COMMON-LISP")
                     (SETQ MODENAME "LISP"))
                 (SETQ OBJECT (IL:FIND-READTABLE MODENAME))
                 (COND
                    ((READTABLEP OBJECT)
                     ;; Need to continue intercepting comments!
                     (SETQ *READTABLE* (COPY-READTABLE OBJECT))
                     (IL:READTABLEPROP *READTABLE* 'IL:NAME (IL:READTABLEPROP OBJECT 'IL:NAME))
```

```
                    (SET-MACRO-CHARACTER #\; #'CLFILE-READ-SEMI NIL *READTABLE*))
                   (T (ERROR "~&Non-existent readtable: ~A~%" MODENAME))))
             (WHEN (SETQ MODEPOS (SEARCH "BASE: " MODESTR))
                 (SETQ OBJECT (READ-FROM-STRING MODESTR NIL NIL :START (+ MODEPOS (LENGTH "BASE: "))))
                 (COND
                    ((AND (NUMBERP OBJECT)
                          (> OBJECT 0))
                     (SETQ *READ-BASE* (TRUNCATE OBJECT)))
                    (T (ERROR "~&Bad read base: ~A~%" OBJECT))))
             (WHEN FILE-ID
                 (SETF (GET FILE-ID 'IL:MAKEFILE-ENVIRONMENT)
                       (LIST :PACKAGE (PACKAGE-NAME *PACKAGE*)
                             :READTABLE
                             (LET ((RDTBLNAME (IL:READTABLEPROP *READTABLE* 'IL:NAME)))
                                  (COND
                                     ((OR (NOT (STRINGP RDTBLNAME))
                                          (STRING-EQUAL RDTBLNAME "LISP"))
                                      "XCL")
                                     (T RDTBLNAME)))
                             :BASE *READ-BASE*)))
             (RETURN T))))

(DEFUN CLFILE-READ-SEMI (STREAM RDTBL)
   "A ; was seen.  Collect more ;'s, then wrap comment in IL:*"
   (DECLARE (IGNORE RDTBL))
   (DO ((CH (READ-CHAR STREAM)
            (READ-CHAR STREAM))
        (LVL 1)
        (COMMENT ""))
       ((OR (NULL CH)
            (NOT (CHAR= CH #\;)))
        (UNREAD-CHAR CH STREAM)
        (COND
           ((SETQ COMMENT (READ-LINE STREAM))
            (LIST 'IL:* (COND
                           ((> LVL 2)
                            'IL:|;;;|)
                           ((= LVL 2)
                            'IL:|;;|)
                           (T 'IL:\;))
                  (STRING-TRIM '(#\Space #\Tab)
                               COMMENT)))))
     (INCF LVL 1)))

(DEFUN CLFILE-SET-MODE (MAKEFILE-ENVIRONMENT)

;;; Using the MAKEFILE-ENVIRONMENT, sets the appropriate free vars (bound above in EXPORT-CLFILE) to the specified package, readtable and
;;; print base.

   (DECLARE (SPECIAL *PACKAGE* *READTABLE* *PRINT-BASE*))
   (LET ((FILE-PACKAGE (GETF MAKEFILE-ENVIRONMENT :PACKAGE))
         (READ-TABLE (GETF MAKEFILE-ENVIRONMENT :READTABLE))
         (PRINT-BASE (GETF MAKEFILE-ENVIRONMENT :BASE)))
        (LET ((PKG (IF (STRINGP FILE-PACKAGE)
                       (FIND-PACKAGE FILE-PACKAGE)
                       (EVAL FILE-PACKAGE))))
             (WHEN (PACKAGEP PKG)
                   (SETQ *PACKAGE* PKG)))
        (LET ((RDTBL (IF (STRINGP READ-TABLE)
                         (IL:FIND-READTABLE READ-TABLE)
                         (EVAL READ-TABLE))))
             (WHEN (READTABLEP RDTBL)
                   (SETQ *READTABLE* RDTBL)))
        (LET ((BASE (IF (NUMBERP PRINT-BASE)
                        PRINT-BASE
                        (EVAL PRINT-BASE))))
             (WHEN (NUMBERP BASE)
                   (SETQ *PRINT-BASE* BASE)))))

(IL:PUTPROPS IL:PORT-CLFILE IL:FILETYPE :COMPILE-FILE)

(IL:PUTPROPS IL:PORT-CLFILE IL:MAKEFILE-ENVIRONMENT (:READTABLE "XCL" :PACKAGE "XCL" :BASE 10))

(IL:PUTPROPS IL:PORT-CLFILE IL:COPYRIGHT ("Johannes Koomen, Larry Masinter" 1987))
```

FUNCTION INDEX

PROPERTY INDEX