File created: 13-Jul-88 17:58:45 **{POGO:AISNORTH:XEROX}<LOOPSCORE>USERS>CONVERSION-AIDS.;5**

changes to: (FUNCTIONS DEFINE-METHOD-FNS)

previous date: 16-Jun-88 12:20:55 {POGO:AISNORTH:XEROX}<LOOPSCORE>USERS>CONVERSION-AIDS.;4

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
```


```
(RPAQQ CONVERSION-AIDSCOMS
       ((DECLARE%: DONTCOPY (PROP MAKEFILE-ENVIRONMENT CONVERSION-AIDS))

;;; Some aids for converting files from KOTO-Loops to Lyric-Loops

        (FNS CONVERT-BQUOTE CONVERT-COMMENTS CONVERT-COMS CONVERT-METHODS CONVERT-ONE-COMMENT
             CONVERT-LOOPS-FILES)
        (FUNCTIONS DEFINE-METHOD-FNS WITH-LOOP-LIMIT)
        (INITVARS (SEDIT.CONVERT.UPGRADE 100))
                                                        ; Different name under Medley, so set it if not already.
        ))

(DECLARE%: DONTCOPY

(PUTPROPS CONVERSION-AIDS MAKEFILE-ENVIRONMENT (:PACKAGE "IL" :READTABLE "INTERLISP"))
)

;;; Some aids for converting files from KOTO-Loops to Lyric-Loops

(DEFINEQ

(CONVERT-BQUOTE
  [LAMBDA (NAME TYPE)                                   ; Edited 16-Jun-88 12:08 by raf

;;; Fix up old style BQUOTE forms left over from a KOTO file

    (if (HASDEF NAME TYPE)
        then (WITH-LOOP-LIMIT 1000 (EDITDEF NAME TYPE NIL
                                            (for pair in '((%, . \,)
                                                           (%,@ . \,@)
                                                           (%,. . \,.))
                                               join `(^ (LPQ (F ,(CAR pair)
                                                                1)
                                                             (BI 1 2)
                                                             1
                                                             (REPLACE 1 WITH ,(CDR pair])

(CONVERT-COMMENTS
  [LAMBDA (NAME TYPE)                                   ; Edited  1-Sep-87 17:57 by smL

;;; Convert the  top-level comments to the new-style comment

    (if (HASDEF NAME TYPE)
        then (WITH-LOOP-LIMIT 300 (EDITDEF NAME TYPE NIL
                                           `((LPQ (F (,COMMENTFLG --)
                                                     NIL)
                                                  UP
                                                  [COMS `(REPLACE 1 WITH ,(CONVERT-ONE-COMMENT (CAR (%##]
                                                  2 UP])

(CONVERT-COMS
  [LAMBDA (FILE-NAME)                                   ; Edited 16-Jun-88 12:09 by raf

;;; Change the COMS of a file so that any DEFMACRO forms are now dumped out as (FUNCTIONS ...) instead.

    (WITH-LOOP-LIMIT
     300
     (EDITDEF FILE-NAME 'FILES NIL
            '((LPQ (F (MACROS --)
                      1)
                   [COMS (LET* ((FUNCTION-NAMES (for m in (CDR (%##)) when (HASDEF m 'FUNCTIONS) collect m))
                                (MACRO-NAMES (LDIFFERENCE (CDR (%##))
                                                          FUNCTION-NAMES)))
                               (if (NULL FUNCTION-NAMES)
                                   then NIL
                                 elseif (NULL MACRO-NAMES)
                                   then '(REPLACE 1 WITH FUNCTIONS)
```

```
                                    else `(COMSQ UP (REPLACE 1 WITH (COMS (MACROS ,@MACRO-NAMES)
                                                                          (FUNCTIONS ,@FUNCTION-NAMES]
                        (ORR (NX)
                             (!NX)
                             NIL])
```

(CONVERT-METHODS

```
  [LAMBDA (FILE)                                                    ; Edited 28-Sep-87 11:10 by smL
```

;;; Scan the file, converting any (DEFINEQ ...) expressions that define methods into (DEFINE-METHOD-FNS ...), so the file can be loaded

```
    (CL:WITH-OPEN-STREAM (TS (OPENTEXTSTREAM (MKATOM FILE)))
           (for pos in (REVERSE (CL:WITH-OPEN-FILE (S FILE :DIRECTION :INPUT)
                                      (bind (*PACKAGE* _ (CL:FIND-PACKAGE "IL"))
                                            (*READTABLE* _ (FIND-READTABLE "OLD-INTERLISP-FILE"))
                                            (EOF _ (CONS NIL NIL))
                                           pos form eachtime (SETQ pos (GETFILEPTR S))
                                                             (SETQ form (CL:READ S NIL EOF))
                                       until (OR (EQ form EOF)
                                                 (EQ form 'STOP))
                                       when (AND (EQ (CL:FIRST form)
                                                     'DEFINEQ)
                                                 (EQ (GETPROP (CL:FIRST (CL:SECOND (CL:SECOND form)))
                                                              'DEFINER-FOR)
                                                     'METHOD-FNS))
                                       collect pos)))
                 bind (file-changed-p _ NIL) do (SETQ pos (TEDIT.FIND TS "DEFINEQ" pos))
                                                (TEDIT.DELETE TS pos 7)
                                                (TEDIT.INSERT TS "DEFINE-METHOD-FNS" pos)
                                                (SETQ file-changed-p T)
                 finally (if file-changed-p
                             then (TEDIT.PUT TS FILE NIL T)
                                  (RETURN T)
                           else (RETURN NIL])
```

(CONVERT-ONE-COMMENT

```
  [LAMBDA (comment)                                                 ; Edited 11-Jun-87 13:23 by smL
    (if (NOT (AND (LISTP comment)
                  (EQ (CAR comment)
                      COMMENTFLG)))
        then                                                        ; Not a comment, so nothing to convert
             comment
      elseif (MEMB (CADR comment)
                   '(; ;; ;;;))
        then                                                        ; Already converted
             comment
      else (LET [(comment-string (MKSTRING (if (EQ (CADR comment)
                                                   COMMENTFLG)
                                               then (CDDR comment)
                                               else (CDR comment]   ; Strip off the leading and trailing parens
                 [SETQ comment-string (SUBSTRING comment-string 2 (SUB1 (NCHARS comment-string]
                 `(,COMMENTFLG ,(if (EQ (CADR comment)
                                        COMMENTFLG)
                                    then                            ; Major comment
                                         ';;;
                                  elseif (GEQ (NCHARS comment-string)
                                              SEDIT.CONVERT.UPGRADE)
                                    then                            ; Semi-major comment
                                         ';;
                                  else                              ; Minor comment
                                       ';)
                                ,comment-string])
```

(CONVERT-LOOPS-FILES

```
  [LAMBDA (files dump-files-p)                                      ; Edited 16-Jun-88 12:13 by raf
```

;; First, make a pass thru all the files, editing them so they can be loaded. That way, if any of these files refer to any other of the files, there will ;; already be a loadable version sitting around.

```
    (for file inside files do (PRINTOUT T "Converting methods on file " file T)
                              (CONVERT-METHODS file))
```

;; Now load all the files (see above note).

```
    [for file inside files do (PRINTOUT T "Loading file " file T)
                              (LET [(ORIGINAL-SYNTAX (GETSYNTAX LoopsReadMacroChar (FIND-READTABLE
                                                                                    "OLD-INTERLISP-FILE"]
                                   (CL:UNWIND-PROTECT
                                       [PROGN (SETSYNTAX LoopsReadMacroChar '(MACRO FIRST HashMacro)
                                                         (FIND-READTABLE "OLD-INTERLISP-FILE"))
                                              (DOFILESLOAD `((SOURCE)
                                                             ,file]
                                       (SETSYNTAX LoopsReadMacroChar ORIGINAL-SYNTAX (FIND-READTABLE
                                                                                      "OLD-INTERLISP-FILE"))
)]
```

;; Give them all explicit makefile-environments

```
[for file inside files do (PRINTOUT T "Defining makefile-environment for file " file T)
                          (PUTPROP (ROOTFILENAME file)
                                 'MAKEFILE-ENVIRONMENT
                                 '(:PACKAGE "IL" :READTABLE "INTERLISP" :BASE 10))
                          (SET (FILECOMS file)
                               `([DECLARE%: DONTCOPY (PROP MAKEFILE-ENVIRONMENT ,(ROOTFILENAME file]
                                 ,@(EVALV (FILECOMS file]
```

;; Walk thru the files, cleaning up any odd forms

```
[for file inside files
   do (BLOCK)
      ;; Change any DEFMACRO forms from MACROS to FUNCTIONS on the coms of the file
      (PRINTOUT T "Changing DEFMACROs to FUNCTIONS in the coms of file " file T)
      (CONVERT-COMS (ROOTFILENAME file))
      ;; Fix any odd BQUOTE forms that are lying around
      (PRINTOUT T "Fixing old BQUOTE expressions" T)
      (CONVERT-BQUOTE (ROOTFILENAME file)
            'FILES)
      (for type in '(MACROS FNS FUNCTIONS) do (for m in (FILECOMSLST (ROOTFILENAME file)
                                                                    type)
                                                      do (BLOCK)
                                                         (CONVERT-BQUOTE m type)))
      ;; Fix comments
      (PRINTOUT T "Upgrading some comments" T)
      (CONVERT-COMMENTS file 'FILES)
      (for type in '(FUNCTIONS) do (for m in (FILECOMSLST (ROOTFILENAME file)
                                                          type)
                                         do (BLOCK)
                                            (CONVERT-COMMENTS m type)))
      ;; Clean up the documentation in the classes...
      (PRINTOUT T "Converting documentation comments to strings in classes" T)
      (for class-name in (FILECOMSLST (ROOTFILENAME file)
                                      'CLASSES)
         do (BLOCK)
            (if (HASDEF class-name 'CLASSES)
                then (MARKASCHANGED class-name 'CLASSES)
                     [LET ((c ($! class-name)))
                          [LET* ((doc (GetClass c 'doc))
                                 (new-doc (CONVERT-ONE-COMMENT doc)))
                                (if (NEQ new-doc doc)
                                    then (PutClass c (CAR (LAST new-doc))
                                                   'doc]
                          [for iv in (_ c ListAttribute 'IVS)
                             do (LET* ((doc (GetValue c iv 'doc))
                                       (new-doc (CONVERT-ONE-COMMENT doc)))
                                      (if (NEQ new-doc doc)
                                          then (PutValue c iv (CAR (LAST new-doc))
                                                         'doc]
                          (for cv in (_ c ListAttribute 'CVS)
                             do (LET* ((doc (GetClassValue c cv 'doc))
                                       (new-doc (CONVERT-ONE-COMMENT doc)))
                                      (if (NEQ new-doc doc)
                                          then (PutClassValue c cv (CAR (LAST new-doc))
                                                              'doc]
                else (DELFROMFILE class-name 'CLASSES file)))
      ;; ... and the methods
      (PRINTOUT T "Converting documentation comments to strings in methods " T)
      (for method-name in (FILECOMSLST (ROOTFILENAME file)
                                       'METHODS)
         do (BLOCK)
            (if (HASDEF method-name 'METHODS)
                then [LET* ((method-obj ($! method-name))
                            (doc (GetValue method-obj 'doc))
                            (new-doc (CONVERT-ONE-COMMENT doc)))
                           (if (NEQ new-doc doc)
                               then (PutValue method-obj 'doc (CAR (LAST new-doc)))
                                    (MARKASCHANGED method-name 'METHODS]
                else (DELFROMFILE method-name 'METHODS]
```

;; Dump out the files

```
(SELECTQ dump-files-p
     (:COMPILE (for file inside files do (PRINTOUT T "Dumping & Compiling file " file T)
                                        (MAKEFILE file '(NEW))
                                        (CL:COMPILE-FILE file)))
     (:COMPILE/LOAD
          (for file inside files do (PRINTOUT T "Dumping, Compiling, & Loading file " file T)
                                    (MAKEFILE file '(NEW))
                                    (LOAD (CL:COMPILE-FILE file))))
     (if dump-files-p
         then (for file inside files do (PRINTOUT T "Dumping file " file T)
                                        (MAKEFILE file '(NEW])
```

```
)

(DEFMACRO DEFINE-METHOD-FNS (&REST METHOD-DEFS)
    ;; This is uglier than it should be, because of the need to convert old-style top-level comments in the body of the method.  Otherwise, evaling the
    ;; result produces nasty warnings about "Possible comment not stripped"
    [CONS 'PROGN
          (for def in METHOD-DEFS
             collect (COND
                       [(IGNORE-ERRORS (LET ((FULL-METHOD-NAME (CL:FIRST def))
                                             (METHOD-WORD (CL:FIRST (CL:SECOND def)))
                                             METHOD-KEYWORDS METHOD-ARGLIST METHOD-BODY CLASS-NAME SELECTOR ARGS)
                                            ;; Parse the def, which looks like (Foo.Bar (Method {keyword value}* ((ClassName Selector)
                                            ;; ..args) ..forms))
                                            [SETQ METHOD-KEYWORDS (for x on (CL:REST (CL:SECOND def)) by CDDR
                                                                     until (LISTP (CL:FIRST x))
                                                                     join `(,(CL:FIRST x)
                                                                            ,(CL:SECOND x))
                                                                     finally (SETQ METHOD-ARGLIST (CL:FIRST x))
                                                                             (SETQ METHOD-BODY (CL:REST x]
                                            (SETQ CLASS-NAME (CL:FIRST (CL:FIRST METHOD-ARGLIST)))
                                            (SETQ SELECTOR (CL:SECOND (CL:FIRST METHOD-ARGLIST)))
                                            (SETQ ARGS (CL:REST METHOD-ARGLIST))
                                            ;; Sometimes Loops got confused when renaming a method, and the CLASS-NAME and SELECTOR
                                            ;; don't jive with the real method name.  So we have to make sure that they are OK
                                            [if (NOT (EQ FULL-METHOD-NAME (PACK* CLASS-NAME "." SELECTOR)))
                                                then                ; OK gang, we got a problem
                                                     (LET ((delimiter-pos (STRPOS "." FULL-METHOD-NAME)))
                                                          (if (EQP delimiter-pos
                                                                   (STRPOS "." FULL-METHOD-NAME NIL NIL NIL NIL NIL
                                                                           T))
                                                              then ; Only one delimeter, so se know how to break it in two
                                                                   (SETQ CLASS-NAME (SUBATOM FULL-METHOD-NAME 1
                                                                                             (SUB1 delimiter-pos)))
                                                                   (SETQ SELECTOR (SUBATOM FULL-METHOD-NAME
                                                                                           (ADD1 delimiter-pos)))
                                                            else    ; HELP!
                                                                 (CL:WARN "~Method name ~S doesn't match
                                                                      method-body" FULL-METHOD-NAME]
                                            ;; Convert top-level comments in the body of the method, to keep the reader from generating
                                            ;; lots of awful warning messages
                                            ;; JRB - first find the documentation comment, if any, and make it a string
                                            [LET ((DOC-COMMENT-INDEX 0)
                                                  FOUND-ONE)
                                                 (while (AND (LISTP (CL:NTH DOC-COMMENT-INDEX METHOD-BODY))
                                                             (EQ (CAR (CL:NTH DOC-COMMENT-INDEX METHOD-BODY))
                                                                 COMMENTFLG))
                                                    do (CL:INCF DOC-COMMENT-INDEX)
                                                       (SETQ FOUND-ONE T))
                                                 (CL:DECF DOC-COMMENT-INDEX)
                                                 ;; HACK: If there is a string in the selected comment already, it probably is the date
                                                 ;; stamp and shouldn't be mucked with
                                                 (if (AND FOUND-ONE (for X in (CL:NTH DOC-COMMENT-INDEX METHOD-BODY)
                                                                       never (STRINGP X)))
                                                     then (CL:SETF (CL:NTH DOC-COMMENT-INDEX METHOD-BODY)
                                                                   (CADDR (CONVERT-ONE-COMMENT (CL:NTH
                                                                                                   DOC-COMMENT-INDEX
                                                                                                   METHOD-BODY]
                                            (SETQ METHOD-BODY (for x in METHOD-BODY
                                                                 collect (if (AND (LISTP x)
                                                                                  (EQ (CAR x)
                                                                                      COMMENTFLG))
                                                                             then (CONVERT-ONE-COMMENT x)
                                                                           else x)))
                                            ;; Got it all figured out...
                                            `(,METHOD-WORD ,@METHOD-KEYWORDS ((,CLASS-NAME ,SELECTOR)
                                                                              ,@ARGS)
                                                           ,@METHOD-BODY]
                                       (T (CL:FORMAT *ERROR-OUTPUT* "Trouble converting method ~A:~%if this is a RuleSet, you
                                                 must recompile it~%" (if (CL:CONSP def)
                                                                          then (CAR def)
                                                                        else def))
                                          NIL])

(DEFMACRO WITH-LOOP-LIMIT (VALUE &BODY FORMS)
   [LET ((VAR (GENSYM)))
        `(LET ((,VAR MAXLOOP))
              (CL:UNWIND-PROTECT
                  (PROGN (SETQ MAXLOOP ,VALUE)
                         ,@FORMS)
```

```
                    (SETQ MAXLOOP ,VAR))])

(RPAQ? SEDIT.CONVERT.UPGRADE 100)
```

;; Different name under Medley, so set it if not already.

```
(PUTPROPS CONVERSION-AIDS COPYRIGHT ("Xerox Corporation" 1987 1988))
```

FUNCTION INDEX

MACRO INDEX

VARIABLE INDEX

PROPERTY INDEX