File created: 12-Oct-93 22:11:23 **{Pele:mv:envos}<LispCore>Sources>CLTL2>LLREAD.;2**

changes to: (FNS \RSTRING2)
(MACROS \NSIN)

previous date: 4-May-91 08:47:17 {Pele:mv:envos}<LispCore>Sources>CLTL2>LLREAD.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;


```
(RPAQQ LLREADCOMS
       [(COMS                                                    ; Reader entrypoints
              (FNS LASTC PEEKC PEEKCCODE RATOM READ READC READCCODE READP SETREADMACROFLG SKIPSEPRCODES
                   SKIPSEPRS \NSIN.24BITENCODING.ERROR SKREAD))
        (COMS                                                    ; CommonLisp read entry points
              (FNS CL:READ CL:READ-PRESERVING-WHITESPACE CL:READ-DELIMITED-LIST CL:PARSE-INTEGER))
        (COMS                                                    ; reading strings
              (FNS RSTRING READ-EXTENDED-TOKEN \RSTRING2))
        (COMS                                                    ; Core of the reader
              (FNS \TOP-LEVEL-READ \SUBREAD \SUBREADCONCAT \SUBREAD-FLIPCASE READTABLE-CASEARRAY \READ.SYMBOL
                   \INVALID.SYMBOL \APPLYREADMACRO INREADMACROP))
        (COMS                                                    ; Read macro for '
              (FNS READQUOTE))
        (COMS                                                    ; # macro
              (FNS READVBAR READHASHMACRO DEFMACRO-LAMBDA-LIST-KEYWORD-P DIGITBASEP READNUMBERINBASE
                   ESTIMATE-DIMENSIONALITY SKIP.HASH.COMMENT CMLREAD.FEATURE.PARSER))
        (COMS                                                    ; Reading characters with #\
              (FNS CHARACTER.READ CHARCODE.DECODE)
              (VARS CHARACTERNAMES CHARACTERSETNAMES))
        (DECLARE%: DOEVAL@COMPILE DONTCOPY (CONSTANTS * READTYPES)
               (MACROS .CALL.SUBREAD. FIXDOT RBCONTEXT PROPRB \RDCONC)
               (EXPORT (MACROS \BACKCHAR \BACKNSCHAR \CHECKEOLC \INCHAR \INCCODE \PEEKCCODE \NSIN \NSPEEK
                               NUMERIC-CHARSET))
               (SPECVARS *READ-NEWLINE-SUPPRESS* \RefillBufferFn)
               (GLOBALVARS *KEYWORD-PACKAGE* *INTERLISP-PACKAGE*)
               (RESOURCES \PNAMECASEBITS)
                                                                 ; For READTABLE-CASE :INVERT
               )
        [COMS                                                    ; Support for various external formats
              [COMS                                              ; JIS to XCCS conversion table.
                    (VARS *JIS-TO-XCCS-CONV-NO-FONT-TABLE* *JIS-TO-XCCS-CODE-MAP* *HANKAKU-TO-ZENKAKU-CODE-MAP*)
                    (GLOBALVARS *JIS-TO-XCCS-CONV-NO-FONT-TABLE* *JIS-TO-XCCS-CONV-TABLE-LIST*
                           *JIS-TO-XCCS-CODE-MAP* *HANKAKU-TO-ZENKAKU-CODE-MAP* *JIS-1KU-TO-XCCS-CONV-TABLE*
                           *JIS-2KU-TO-XCCS-CONV-TABLE* *JIS-6KU-TO-XCCS-CONV-TABLE* *XCCS-TO-JIS-CONV-TABLE*
                           *HANKAKU-TO-ZENKAKU-CONV-TABLE* *ZENKAKU-TO-HANKAKU-CONV-TABLE*)
                    (FNS \MAKE.JIS.TO.XCCS.CONV.TABLE)
                    (DECLARE%: DONTEVAL@LOAD DOCOPY (P (\MAKE.JIS.TO.XCCS.CONV.TABLE]
              [COMS                                              ; JIS to XCCS converter
                    (INITVARS (*REPLACE-NO-FONT-CODE* T)
                           (*DEFAULT-NOT-CONVERTED-FAT-CODE* 8739))
                    (GLOBALVARS *REPLACE-NO-FONT-CODE* *DEFAULT-NOT-CONVERTED-FAT-CODE*)
                    (DECLARE%: DOEVAL@COMPILE DONTCOPY (EXPORT (MACROS \CONV.JIS.TO.XCCS \DO.CONV.JIS.TO.XCCS]
              [COMS                                              ; XCCS to JIS converter
                    (FNS CONVHANKAKU)
                    (DECLARE%: DOEVAL@COMPILE DONTCOPY (EXPORT (MACROS \CONV.XCCS.TO.JIS \DO.CONV.XCCS.TO.JIS
                                                                       \ASCIIP \NOT.EQUIVALENT.TO.JIS
                                                                       \CONV.HANKAKU.TO.ZENKAKUP
                                                                       \CONV.ZENKAKU.KANA]
              (COMS (FNS \JISIN \JISPEEK \BACKJISCHAR \SHIFTJISIN \SHIFTJISPEEK \BACKSHIFTJISCHAR \EUCIN
                         \EUCPEEK \BACKEUCCHAR \THROUGHIN \THROUGHPEEK \BACKTHROUGHCHAR)
                    (DECLARE%: DOEVAL@COMPILE DONTCOPY
                           (EXPORT
                                  ;; XCCS specific macro.  Although the decoder and encoder are implemented as functions in general, only
                                  ;; for XCCS, they are implemeted as macros for efficiency reason.
                                  (MACROS \XCCSIN \XCCSPEEK \BACKXCCSCHAR \XCCSP)
                                  ;; JIS specific macro
                                  (MACROS \EXTRACT.NO.FONT.CODE \EXTARACT.CONV.TABLE \NOT.EQUIVALENT.TO.XCCS
                                         \EXTRACT.SET \EXTRACT.CODE \CHNAGE.KI.MODE \KIMODEP \HANKAKUP \KANJIP
                                         \NOTGAIJIP \INVALID.TENP \CONV.HANKAKU.KANA \OUTKI \OUTKO)
                                  ;; Shift-JIS specific macro
                                  (MACROS \CONV.SJIS.TO.JIS \CONV.JIS.TO.SJIS \SJIS.KANJI.FIRST.BYTEP)
                                  ;; EUC specific macro
                                  (MACROS \EUC.KANJI.FIRST.BYTEP \GAIJIP \EUC.HANKAKUP]
        (INITVARS (*SIGNAL-24BIT-NSENCODING-ERROR*)
               (*READ-NEWLINE-SUPPRESS*)
               (\RefillBufferFn (FUNCTION \READCREFILL)))
```

```
                                                                    ; Top level val of \RefillBufferFn means act like READC--we must
                                                                    ; be doing a raw BIN (or PEEKBIN?)
        (LOCALVARS . T)
        (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                              (NLAML)
                                                                              (LAMA CONVHANKAKU CL:PARSE-INTEGER
                                                                                    CL:READ-DELIMITED-LIST
                                                                                    CL:READ-PRESERVING-WHITESPACE
                                                                                    CL:READ])

;; Reader entrypoints

(DEFINEQ

(LASTC
  [LAMBDA (FILE)                                                    ; Edited  6-Jan-88 15:31 by jds

    ;; Be careful only to do BIN's if we first were able to back up, so that an EOF doesn't happen.  This is really an inadequate implementation,
    ;; because it fails for files that cannot be backed up.  Eventually, we must change the character reading functions READ, RATOM, READC to save
    ;; the last character they read in an STREAM field.

    (LET* ((STREAM (\GETSTREAM FILE 'INPUT))
           (LASTCCODE (FETCH (STREAM LASTCCODE) OF STREAM)))

          ;; (FCHARACTER (SELCHARQ C (CR (SELECTC (ffetch EOLCONVENTION of STREAM) (CR.EOLC (CHARCODE EOL)) C)) (LF
          ;; (SELECTC (ffetch EOLCONVENTION of STREAM) (LF.EOLC (CHARCODE EOL)) (CRLF.EOLC (COND ((EQ (CHARCODE CR)
          ;; (UNINTERRUPTABLY (AND (\BACKNSCHAR STREAM SHIFTEDCHARSET) (PROG1 (PROGN (\BACKNSCHAR STREAM
          ;; SHIFTEDCHARSET) (\NSIN STREAM SHIFTEDCHARSET)) (\NSIN STREAM SHIFTEDCHARSET)))))) (CHARCODE EOL)) (T C))) C))
          ;; (NIL 0) C))

          (COND
             ((IEQP LASTCCODE 65535)
              NIL)
             (T (FCHARACTER LASTCCODE])

(PEEKC
  [LAMBDA (FILE FLG)                                                (* rmk%: "10-Apr-85 11:55")

    ;; FLG says to proceed as if Control were T--not implemented correctly here NIL

    (LET [(\RefillBufferFn (FUNCTION \PEEKREFILL))
          (STREAM (\GETSTREAM FILE 'INPUT]
         (DECLARE (SPECVARS \RefillBufferFn))
         (FCHARACTER (PEEKCCODE STREAM])

(PEEKCCODE
  [LAMBDA (FILE NOERROR)                                            (* bvm%: "12-Sep-86 15:19")
    (LET [(\RefillBufferFn (FUNCTION \PEEKREFILL))
          (STREAM (\GETSTREAM FILE 'INPUT]
         (DECLARE (SPECVARS \RefillBufferFn))
         (\PEEKCCODE STREAM NOERROR])

(RATOM
  [LAMBDA (FILE RDTBL)                                              ; Edited  3-Apr-91 21:08 by jrb:

;;; Like READ except interpret break characters as single character atoms.  I.e., always returns an atom

    (SETQ RDTBL (\GTREADTABLE RDTBL))
    (LET ((*READTABLE* RDTBL)
          (*PACKAGE* (if (fetch (READTABLEP USESILPACKAGE) of RDTBL)
                         then *INTERLISP-PACKAGE*
                       else *PACKAGE*))
          (\RefillBufferFn (FUNCTION \RATOM/RSTRING-REFILL)))
         (DECLARE (SPECVARS *READTABLE* *PACKAGE* \RefillBufferFn))
         (.CALL.SUBREAD. FILE NIL NIL NIL T RATOM.RT])

(READ
  [LAMBDA (FILE RDTBL FLG)                                          ; Edited 19-Mar-87 18:35 by bvm:
    (LET ((*READTABLE* (\GTREADTABLE RDTBL))
          (*READ-NEWLINE-SUPPRESS* FLG))
         (DECLARE (SPECVARS *READTABLE* *READ-NEWLINE-SUPPRESS*))

         ;; *READ-NEWLINE-SUPPRESS* is used freely by \FILLBUFFER

         ;; Call reader with PRESERVE-WHITESPACE = T, since that's the semantics Interlisp has always had before (though maybe not explicitly
         ;; stated).

         (\TOP-LEVEL-READ FILE NIL NIL NIL T])

(READC
  [LAMBDA (FILE RDTBL)                                              ; Edited  6-Jan-88 15:30 by jds
    (LET ((*READTABLE* (\GTREADTABLE RDTBL))
          (\RefillBufferFn (FUNCTION \READCREFILL)))
         (DECLARE (SPECVARS *READTABLE* \RefillBufferFn))
         (FCHARACTER (REPLACE (STREAM LASTCCODE) OF (\INSTREAMARG FILE) WITH (\INCCODE (\INSTREAMARG FILE])
```

(READCCODE

```
  [LAMBDA (FILE RDTBL)                                    ; Edited  3-Jun-88 01:30 by atm
```

;;; returns a 16 bit character code. \INCHAR does the EOL conversion and this function converts to a 16 bit value. Saves the character for LASTC as ;;; well.

```
    (SETQ FILE (\GETSTREAM FILE 'INPUT))
    (FDEVOP 'READCHARCODE (fetch (STREAM DEVICE) of FILE)
           FILE RDTBL])
```

(READP

```
  [LAMBDA (FILE FLG)                                      (* rmk%: " 5-Apr-85 09:09")
                                                          ; The 10 does not do the EOL check on the peeked character.
    (LET* ((STREAM (\GETSTREAM FILE 'INPUT))
           (DEVICE (ffetch (STREAM DEVICE) of STREAM)))
          (COND
             ((ffetch (FDEV READP) of DEVICE)
              (FDEVOP 'READP DEVICE STREAM FLG))
             (T (\GENERIC.READP STREAM FLG])
```

(SETREADMACROFLG

```
  [LAMBDA (FLG)                                           (* rmk%: "25-OCT-83 16:13")
                                                          ; D doesn't cause the read-macro context error, hence doesn't
                                                          ; maintain this flag
    NIL])
```

(SKIPSEPRCODES

```
  [LAMBDA (FILE RDTBL)                                    ; Edited  6-Jan-88 13:09 by jds
```

;; Passes over non-separators to peek at the first non-separator on FILE. Returns either last peeked character, or NIL if no non-seprs left in the ;; file.

```
    (bind PREVC C SHIFTEDCHARSET (STREAM _ (\GETSTREAM FILE 'INPUT))
          (SA _ (fetch (READTABLEP READSA) of (\GTREADTABLE RDTBL)))
          (\RefillBufferFn _ '\PEEKREFILL) first (SETQ SHIFTEDCHARSET (UNFOLD (ACCESS-CHARSET STREAM)
                                                                              256))
       declare (SPECVARS \RefillBufferFn) while [EQ SEPRCHAR.RC (\SYNCODE SA (SETQ C
                                                                               (OR (\NSPEEK STREAM SHIFTEDCHARSET
                                                                                            SHIFTEDCHARSET T)
                                                                                   (RETURN]
       do (SETQ PREVC C)
          (\NSIN STREAM SHIFTEDCHARSET SHIFTEDCHARSET)
       finally (AND PREVC (replace (STREAM LASTCCODE) of STREAM with PREVC))
              (RETURN C])
```

(SKIPSEPRS

```
  [LAMBDA (FILE RDTBL)                                    ; Edited 11-Sep-87 17:52 by bvm:
```

;; Passes over non-separators to peek at the first non-separator on FILE. Returns either last peeked character, or NIL if no non-seprs left in the ;; file.

```
    (bind C SHIFTEDCHARSET (STREAM _ (\GETSTREAM FILE 'INPUT))
          (SA _ (fetch (READTABLEP READSA) of (\GTREADTABLE RDTBL)))
          (\RefillBufferFn _ '\PEEKREFILL) first (SETQ SHIFTEDCHARSET (UNFOLD (ACCESS-CHARSET STREAM)
                                                                              256))
       declare (SPECVARS \RefillBufferFn) while [EQ SEPRCHAR.RC (\SYNCODE SA (SETQ C
                                                                               (OR (\NSPEEK STREAM SHIFTEDCHARSET
                                                                                            SHIFTEDCHARSET T)
                                                                                   (RETURN]
       do (\NSIN STREAM SHIFTEDCHARSET SHIFTEDCHARSET) finally (RETURN (FCHARACTER C])
```

(\NSIN.24BITENCODING.ERROR

```
  [LAMBDA (STREAM)                                        (* bvm%: "12-Mar-86 15:35")
    (DECLARE (USEDFREE *SIGNAL-24BIT-NSENCODING-ERROR*))
```

;;; Called if we see the sequence shift,shift on STREAM -- means shift to 24-bit character set, which we don't support. Usually this just means we're ;;; erroneously reading a binary file as text. If this function returns, its value is taken as a character set to shift to

```
    (COND
       (*SIGNAL-24BIT-NSENCODING-ERROR*                   ; Only cause error if user/reader cares
              (ERROR "24-bit NS encoding not supported" STREAM)))
                                                          ; Return charset zero
    0])
```

(SKREAD

```
  [LAMBDA (FILE REREADSTRING RDTBL)                       ; Edited  6-Apr-88 11:06 by amd
    (LET ((*READ-SUPPRESS* 'SKREAD)
          (*READTABLE* (\GTREADTABLE RDTBL))
          (\RBFLG)
          (STRM (\GETSTREAM FILE 'INPUT))
          CH)
         (DECLARE (CL:SPECIAL *READTABLE* *READ-SUPPRESS* \RBFLG))
```

```
        [COND
           (REREADSTRING                                          ; REREADSTRING is string of chars already read.
                  (SETQ STRM (CL:MAKE-CONCATENATED-STREAM (CL:MAKE-STRING-INPUT-STREAM (MKSTRING REREADSTRING))
                                 STRM]                            ; Because of return requirements, have to preview stream for
                                                                  ; unbalanced closing bracket/paren
        (if (NULL (SETQ CH (SKIPSEPRCODES STRM)))
            then (\EOF.ACTION STRM)
          else (SELECTC (PROG1 (\SYNCODE (fetch (READTABLEP READSA) of *READTABLE*)
                                       CH)
                               ;; Read in suppressed mode.  Reader sets \Rbflg free if read ended on unbalanced bracket.  Reason we do the
                               ;; READ in all cases is so that we need to consume the unbalanced paren/bracket, just as if we really had read it;
                               ;; however, READ doesn't set \Rbflg for these cases
                               (\TOP-LEVEL-READ STRM NIL NIL NIL T))
                       (RIGHTPAREN.RC                             ; unbalanced right paren
                           '%))
                       (RIGHTBRACKET.RC                           ; unbalanced right bracket
                           '%])
                       (AND \RBFLG '%]])

)

;; CommonLisp read entry points

(DEFINEQ

(CL:READ
  [CL:LAMBDA (&OPTIONAL (INPUT-STREAM *STANDARD-INPUT*)
                        (EOF-ERROR-P T)
                        EOF-VALUE RECURSIVE-P)                    ; Edited 14-Dec-86 18:48 by bvm
        (COND
           (RECURSIVE-P                                           ; Dive straight into reader using current settings of everything
                  (.CALL.SUBREAD. INPUT-STREAM))
           (T (\TOP-LEVEL-READ INPUT-STREAM (NOT EOF-ERROR-P)
                        EOF-VALUE])

(CL:READ-PRESERVING-WHITESPACE
  [CL:LAMBDA (&OPTIONAL (STREAM *STANDARD-INPUT*)
                        (EOF-ERRORP T)
                        (EOF-VALUE NIL)
                        (RECURSIVEP NIL))                         ; Edited 19-Mar-87 18:33 by bvm:
        ;; Reads from stream and returns the object read, preserving the whitespace that followed the object.
        (COND
           (RECURSIVEP                                            ; Dive straight into reader using current settings of everything
                  (.CALL.SUBREAD. STREAM))
           (T (\TOP-LEVEL-READ STREAM (NOT EOF-ERRORP)
                        EOF-VALUE NIL T])

(CL:READ-DELIMITED-LIST
  [CL:LAMBDA (CHAR &OPTIONAL (INPUT-STREAM *STANDARD-INPUT*)
                        RECURSIVE-P)                              ; Edited 14-Dec-86 18:48 by bvm

;;; Read a list of elements terminated by CHAR.  CHAR must not be a separator char, and ideally should not be a constituent char (if it is, it must be
;;; preceded by whitespace for READ-DELIMITED-LIST to work)

        (LET [(ENDCODE (OR (FIXP CHAR)
                           (CL:CHAR-CODE CHAR)))
              (INSTREAM (\GETSTREAM INPUT-STREAM 'INPUT]
             (if RECURSIVE-P
                 then                                             ; Have to dive into reader without disturbing
                                                                  ; *CIRCLE-READ-LIST*
                      (.CALL.SUBREAD. INPUT-STREAM NIL NIL ENDCODE)
               else (\TOP-LEVEL-READ INPUT-STREAM NIL NIL ENDCODE])

(CL:PARSE-INTEGER
  [CL:LAMBDA (STRING &KEY START END (RADIX 10)
                        JUNK-ALLOWED)                             ; Edited  8-Feb-91 13:24 by gadener
        (CL:IF (NOT (CL:STRINGP STRING))
            (ERROR "This is not a string : ~S" STRING)
            (PROG ((SA (fetch (READTABLEP READSA) of CMLRDTBL))
                   (BASE (fetch (STRINGP BASE) of STRING))
                   (LEN (fetch (STRINGP LENGTH) of STRING))
                   (OFFST (fetch (STRINGP OFFST) of STRING))
                   (FATP (fetch (STRINGP FATSTRINGP) of STRING))
                   MAXDIGITCODE MAXALPHACODE INDEX STOP CHAR SIGN STARTINT ENDINT ERR)
                  (SETQ RADIX (\CHECKRADIX RADIX))
                  (SETQ INDEX (+ OFFST (if (NULL START)
                                           then 0
                                         elseif (< START 0)
                                           then (\ILLEGAL.ARG START)
                                         else START)))
                  (SETQ STOP (+ OFFST (if (NULL END)
                                          then LEN
```

```
                                    elseif (OR (> END LEN)
                                               (< END 0))
                                      then (\ILLEGAL.ARG END)
                                    else END)))
                (SETQ MAXDIGITCODE (+ (CHARCODE 0)
                                      RADIX -1))
                (SETQ MAXALPHACODE (AND (> RADIX 10)
                                        (+ (CHARCODE A)
                                           RADIX -11)))
                (while (AND (< INDEX STOP)
                            (EQ (\SYNCODE SA (\GETBASECHAR FATP BASE INDEX))
                                SEPRCHAR.RC))
                   do                                          ; Skip over separators
                      (SETQ INDEX (CL:1+ INDEX)))
                [COND
                   ((>= INDEX STOP)                            ; no characters remain
                    (RETURN (COND
                               (JUNK-ALLOWED                   ; don't error
                                      (CL:VALUES NIL STOP))
                               (T (SETQ ERR "No non-whitespace characters in integer string: ~S")
                                  (GO FAIL]
```

;; Start parsing a number. Allowed to start with a single sign, then digits in radix, nothing else. Assume collating sequence is (+, -) < ;; digits < uppercase letters < lowercase letters.

```
                (do (SETQ CHAR (\GETBASECHAR FATP BASE INDEX))
                    (if (<= CHAR MAXDIGITCODE)
                        then                                   ; sign or digit
                             (if (>= CHAR (CHARCODE 0))
                                 then                          ; digit
                                      (OR STARTINT (SETQ STARTINT INDEX))
                               elseif (AND (NOT SIGN)
                                           (NOT STARTINT))
                                 then                          ; maybe sign. No good if not at start
                                      (SELCHARQ CHAR
                                           (- (SETQ SIGN '-))
                                           (+ (SETQ SIGN '+))
                                           (RETURN))
                               else (RETURN))
                      elseif (AND MAXALPHACODE (<= (if (>= CHAR (CHARCODE "a"))
                                                       then  ; uppercase it first
                                                            (- CHAR (- (CHARCODE "a")
                                                                       (CHARCODE "A")))
                                                     else CHAR)
                                                   MAXALPHACODE))
                        then                                   ; is alphabetic digit
                             (OR STARTINT (SETQ STARTINT INDEX))
                      else (RETURN))
                   repeatwhile (< (add INDEX 1)
                                  STOP))
                (SETQ ENDINT INDEX)
                (RETURN (CL:VALUES (COND
                                   ([AND STARTINT (OR JUNK-ALLOWED (EQ INDEX STOP)
                                                      (do (if (NEQ (\SYNCODE SA CHAR)
                                                                   SEPRCHAR.RC)
                                                              then
                                                              ; junk found
                                                                   (RETURN NIL)
                                                            elseif (EQ (add INDEX 1)
                                                                       STOP)
                                                              then
                                                              ; at end of string, win
                                                                   (RETURN T)
                                                            else (SETQ CHAR (\GETBASECHAR FATP BASE INDEX]
                                    (\MKINTEGER BASE STARTINT ENDINT (EQ SIGN '-)
                                                RADIX FATP))
                                   (JUNK-ALLOWED NIL)
                                   ((NULL STARTINT)
                                    (SETQ ERR "There aren't any digits in this integer string: ~S.")
                                    (GO FAIL))
                                   (T (SETQ ERR "There is junk in this integer string: ~S.")
                                      (GO FAIL)))
                                (- INDEX OFFST)))
              FAIL
                (CL:ERROR ERR (if (OR START END)
                                  then (CL:SUBSEQ STRING (OR START 0)
                                                         (OR END LEN))
                                else STRING))))])
```

)

;; reading strings

```
(DEFINEQ
```

(**RSTRING**

```
  [LAMBDA (FILE RDTBL RSFLG)                                   ; Edited 22-Mar-87 20:53 by bvm:
    (LET ((*READTABLE* (\GTREADTABLE RDTBL))
```

```
          (\RefillBufferFn '\RATOM/RSTRING-REFILL)
          (*READ-SUPPRESS* NIL))
        (DECLARE (SPECVARS *READTABLE* \RefillBufferFn *READ-SUPPRESS*))
```

;; It's not clear that *READ-SUPPRESS* is supposed to affect anything other than calls to READ. So play it safe and force \Rstring2 to
;; really read a string.

```
        (WITH-RESOURCE (\PNAMESTRING)
               (RSTRING2 (\GETSTREAM FILE 'INPUT)
                      (fetch READSA of *READTABLE*)
                      (OR RSFLG T)
                      \PNAMESTRING])
```

(READ-EXTENDED-TOKEN

```
  [LAMBDA (STREAM RDTBL ESCAPE-ALLOWED-P)                    ; Edited 3-Apr-91 21:16 by jrb:
```

;; This is a cross between RSTRING and \SUBREAD. Read a "token" from STREAM, as defined by the Common Lisp reader and the syntax in
;; RDTBL. EOF terminates as well. If ESCAPE-ALLOWED-P is true, escapes are honored and if one appears, a second value of T is returned.
;; Otherwise, escapes are treated as vanilla chars and the caller can barf on them itself if it desires.

```
    (SETQ RDTBL (\GTREADTABLE RDTBL))
    (WITH-RESOURCE (\PNAMESTRING \PNAMECASEBITS)
           (PROG ((CASEBASE (READTABLE-CASEARRAY RDTBL))
                  (PBASE (ffetch (STRINGP XBASE) of \PNAMESTRING))
                  (SHIFTEDCHARSET (UNFOLD (ACCESS-CHARSET STREAM)
                                          256))
                  (SA (fetch READSA of RDTBL))
                  [INVERTCASE (AND (fetch (READTABLEP LOWER/FLIPCASE) of RDTBL)
                                   (NOT (fetch (READTABLEP CASEINSENSITIVE) of RDTBL]
                  J CH SNX ANSLIST ANSTAIL ESCAPE-APPEARED ESCAPING FATSEEN)
                 (AND INVERTCASE (for old J from 16 to 0 bind (CASEBITSBASE _ (ffetch (ARRAYP BASE) of
                                                                                    \PNAMECASEBITS
                                                                                                  ))
                                  do (\PUTBASE CASEBITSBASE J 0)))
                 (SETQ J 0)
              LP (if (\EOFP STREAM)
                     then                                    ; end of file terminates string just like a sepr/break
                          (GO FINISH))
                 (SETQ CH (\NSIN STREAM SHIFTEDCHARSET SHIFTEDCHARSET))
                                                             ; NOTE: This should really be (\CHECKEOLC (\NSIN --) --), but
                                                             ; eol is usually a break or sepr and the \BACKNSCHAR doesn't
                                                             ; work right. Fix this when we unread correctly
                 (SETQ SNX (\SYNCODE SA CH))
                 [COND
                    ((AND ESCAPE-ALLOWED-P (SELECTC SNX
                                               (ESCAPE.RC (SETQ CH (\CHECKEOLC (\NSIN STREAM SHIFTEDCHARSET
                                                                                     SHIFTEDCHARSET)
                                                                               (ffetch EOLCONVENTION of STREAM)
                                                                               STREAM))
                                                          (SETQ ESCAPE-APPEARED T)
                                                          (SETA \PNAMECASEBITS J 1))
                                               (MULTIPLE-ESCAPE.RC
                                                    (SETQ ESCAPING (NOT ESCAPING))
                                                    (SETQ ESCAPE-APPEARED T)
                                                    (GO LP))
                                               NIL)))
                                                             ; eat chars until next |
                    (ESCAPING
                           )
                    ((fetch STOPATOM of SNX)
                     (\BACKNSCHAR STREAM SHIFTEDCHARSET)
                     (GO FINISH))
                    ((AND CASEBASE (ILEQ CH \MAXTHINCHAR))
                     (SETQ CH (\GETBASEBYTE CASEBASE CH]
                 (COND
                    ((EQ J \PNAMELIMIT)                      ; Filled PNSTR so have to save those chars away and start filling
                                                             ; up a new buffer
                     (SETQ J (\SMASHSTRING (ALLOCSTRING J NIL NIL FATSEEN)
                                   0 \PNAMESTRING J))
                     [COND
                        [ANSLIST (RPLACD ANSTAIL (SETQ ANSTAIL (CONS J NIL]
                        (T (SETQ ANSTAIL (SETQ ANSLIST (CONS J NIL]
                     (SETQ J 0)))
                 (\PNAMESTRINGPUTCHAR PBASE J CH)
                 (COND
                    ((AND (NOT FATSEEN)
                          (IGREATERP CH \MAXTHINCHAR))
                     (SETQ FATSEEN T)))
                 (AND INVERTCASE ESCAPING (SETA \PNAMECASEBITS J 1))
                 (SETQ J (ADD1 J))
                 (GO LP)
              FINISH
                 (AND INVERTCASE (NOT ANSLIST)
                      (\SUBREAD-FLIPCASE PBASE \PNAMECASEBITS (SUB1 J)
                             CASEBASE))
                 (SETQ CH (\SMASHSTRING (ALLOCSTRING J NIL NIL FATSEEN)
                                0 \PNAMESTRING J))
                 [COND
                    (ANSLIST (RPLACD ANSTAIL (SETQ ANSTAIL (CONS CH NIL)))
```

```
                    (SETQ CH (CONCATLIST ANSLIST]
              (RETURN (if ESCAPE-APPEARED
                          then                          ; do it this way because multiple values are slow
                              (CL:VALUES CH T)
                        else CH])
```

(\RSTRING2

```
  [LAMBDA (STREAM SA RSFLG PNSTR)                       ; Edited  4-Aug-93 12:38 by sybalskY:MV:ENVOS
```

;;; The main string reader. Reads characters from STREAM according to the syntax table SA and returns a string. PNSTR is an instance of the global ;;; resource \PNAMESTRING, which we can use all to ourselves as a buffer.

;;; If RSFLG is T then the call is from RSTRING, in which case the string is terminated by a break or sepr in SA. If RSFLG is NIL then the string is ;;; terminated by a string delimiter. If RSFLG is SKIP then CR's and the following separator chars are discarded as an otherwise normal string is read

```
    (DECLARE (USEDFREE *READTABLE* *READ-SUPPRESS*))
    (PROG ((EOLC (ffetch EOLCONVENTION of STREAM))
           (PBASE (SELECTQ (SYSTEMTYPE)
                       (VAX PNSTR)
                       (ffetch (STRINGP XBASE) of PNSTR)))
           (SHIFTEDCHARSET (UNFOLD (ACCESS-CHARSET STREAM)
                                   256))
           (J 0)
           EOLCHAR CH SNX ANSLIST ANSTAIL LASTC FATSEEN SKIPPING)
          (SELECTC EOLC
              (CRLF.EOLC (SETQ EOLCHAR (CHARCODE CR)))
              (CR.EOLC (SETQ EOLCHAR (CHARCODE CR)))
              (LF.EOLC (SETQ EOLCHAR (CHARCODE LF)))
              NIL)
      RS2LP
          (SETQ CH (\NSIN STREAM SHIFTEDCHARSET SHIFTEDCHARSET))
          [COND
             ((EQ CH EOLCHAR)
              ;; We just read the stream's EOL character, so we have to turn it into our EOL.  Most places do this with \CHECKEOLC, but we can't
              ;; do that here, because if the eol is CRLF and would terminate the read, \BACKNSCHAR won't work right.
              (COND
                 ([AND (EQ RSFLG T)
                       (fetch STOPATOM of (\SYNCODE SA (CHARCODE CR]
                                                        ; From RSTRING, eol terminates read.  Leave eol in buffer
                  (\BACKNSCHAR STREAM SHIFTEDCHARSET)
                  (GO FINISH))
                 (T (COND
                       ((AND (EQ EOLC CRLF.EOLC)
                             (EQ (\PEEKBIN STREAM T)
                                 (CHARCODE LF)))        ; Eat the LF after the CR
                        (\BIN STREAM)))
                    (SETQ CH (CHARCODE CR]
          (SETQ SNX (\SYNCODE SA CH))
          (SELECTC SNX
              (OTHER.RC                                 ; Normal case, nothing to do
                        )
              (ESCAPE.RC [COND
                            ((fetch ESCAPEFLG of *READTABLE*)
                             (SETQ CH (\CHECKEOLC (\NSIN STREAM SHIFTEDCHARSET SHIFTEDCHARSET)
                                                  EOLC STREAM))
                             (COND
                                ((AND (EQ RSFLG 'SKIP)
                                      (EQ CH (CHARCODE CR)))     ; Strip leading spaces after escaped returns, too, but leave the
                                                                 ; CR in the string
                                 (SETQ SKIPPING 0)
                                 (GO PUTCHAR])
              (SELECTQ RSFLG
                   (NIL                                 ; end check is dbl quote
                        (COND
                           ((EQ SNX STRINGDELIM.RC)     ; Got it
                            (SETQ LASTC CH)
                            (GO FINISH))))
                   (T                                   ; if called from RSTRING, end check is break or sepr, and we
                                                        ; must leave delim in stream
                      (COND
                         ((fetch STOPATOM of SNX)
                          (\BACKNSCHAR STREAM SHIFTEDCHARSET)
                          (GO FINISH))))
                   (SKIP                                ; Like NIL but strip cr's and leading spaces
                         (SELECTC SNX
                              (STRINGDELIM.RC
                                   (SETQ LASTC CH)
                                   (GO FINISH))
                              (SEPRCHAR.RC              ; Assume that CR is a sepr
                                           (COND
                                              [SKIPPING (COND
                                                           ((EQ CH (CHARCODE EOL))
                                                                 ; Multiple CR's while skipping are kept
                                                            (COND
                                                               ((EQ SKIPPING T)
```

```
                                                                    ; Turn previous space back into CR.  Note that J is guaranteed to
                                                                    ; be at least 1
                                                                   (\PNAMESTRINGPUTCHAR PBASE (SUB1 J)
                                                                        CH)
                                                                   (SETQ SKIPPING 0)))
                                                                (GO PUTCHAR))
                                                               (T     ; Continue skipping seprs
                                                                  (GO RS2LP]
                                                      ((EQ CH (CHARCODE EOL))
                                                                    ; Turn CR into space and start skipping seprs
                                                       (SETQ SKIPPING T)
                                                       (SETQ CH (CHARCODE SPACE))
                                                       (GO PUTCHAR))))
                              NIL))
                    (SHOULDNT)))
          (SETQ SKIPPING NIL)
      PUTCHAR
          [COND
             ((NOT *READ-SUPPRESS*)                                 ; Accumulate character
              (COND
                 ((EQ J \PNAMELIMIT)                                ; Filled PNSTR so have to save those chars away and start filling
                                                                    ; up a new buffer
                  (SETQ J (\SMASHSTRING (ALLOCSTRING J NIL NIL FATSEEN)
                                0 PNSTR J))
                  [COND
                     [ANSLIST (RPLACD ANSTAIL (SETQ ANSTAIL (CONS J NIL]
                     (T (SETQ ANSTAIL (SETQ ANSLIST (CONS J NIL]
                  (SETQ J 0)))
              (\PNAMESTRINGPUTCHAR PBASE J CH)
              (SETQ LASTC CH)
              (COND
                 ((AND (NOT FATSEEN)
                       (IGREATERP CH \MAXTHINCHAR))
                  (SETQ FATSEEN T)))
              (SETQ J (ADD1 J]
          (COND
             ((OR (NEQ RSFLG T)
                  (NOT (\EOFP STREAM)))                             ; in RSTRING (RSFLG=T), if we've read something already, then
                                                                    ; end of file terminates string just like a sepr/break
              (GO RS2LP)))
      FINISH
          (AND LASTC (replace (STREAM LASTCCODE) of STREAM with LASTC))
          (RETURN (COND
                     ((NOT *READ-SUPPRESS*)
                      (SETQ J (\SMASHSTRING (ALLOCSTRING J NIL NIL FATSEEN)
                                    0 PNSTR J))
                      (COND
                         (ANSLIST (RPLACD ANSTAIL (SETQ ANSTAIL (CONS J NIL)))
                                (CONCATLIST ANSLIST))
                         (T J])

)
```