File created: 16-May-90 21:47:02 **{DSK}<usr>local>lde>lispcore>sources>SEDIT-COMMENTS.;2**

changes to: (IL:VARS IL:SEDIT-COMMENTSCOMS)

previous date: 27-Apr-88 11:20:49 {DSK}<usr>local>lde>lispcore>sources>SEDIT-COMMENTS.;1

Read Table: XCL

Package: SEDIT

Format: XCCS


```
(IL:RPAQQ IL:SEDIT-COMMENTSCOMS
          ((IL:PROP IL:FILETYPE IL:SEDIT-COMMENTS)
           (IL:PROP IL:MAKEFILE-ENVIRONMENT IL:SEDIT-COMMENTS)
           (IL:LOCALVARS . T)
           (IL:DECLARE\: IL:DONTCOPY IL:DOEVAL@COMPILE (IL:FILES IL:SEDIT-DECLS))
           (IL:CONSTANTS (LEVEL-1-COMMENT 'IL:\;)
                  (LEVEL-2-COMMENT 'IL:|;;|)
                  (LEVEL-3-COMMENT 'IL:|;;;|)
                  (LEVEL-4-COMMENT 'IL:|;;;;|)
                  (LEVEL-5-COMMENT 'IL:\|)
                  (COMMENT-LEVEL-TABLE (LIST LEVEL-1-COMMENT 1 LEVEL-2-COMMENT 2 LEVEL-3-COMMENT 3
                                             LEVEL-4-COMMENT 4 LEVEL-5-COMMENT 5))
                  (COMMENT-MARKERS (LIST LEVEL-1-COMMENT LEVEL-2-COMMENT LEVEL-3-COMMENT LEVEL-4-COMMENT
                                         LEVEL-5-COMMENT)))
           (IL:FNS BACKSPACE-COMMENT CFV-COMMENT CLOSE-NODE-COMMENT COMMENT-LENGTH COMPUTE-COMMENT-COLUMN
                  COMPUTE-POINT-POSITION-COMMENT COMPUTE-SELECTION-POSITION-COMMENT COPY-SELECTION-COMMENT
                  COPY-STRUCTURE-COMMENT COPY-STRUCTURE-COMMENT-WORD CREATE-NEW-COMMENT DEGRADE-COMMENT
                  DELETE-COMMENT INITIALIZE-COMMENTS INSERT-COMMENT SPLIT-COMMENT INSERT-COMMENT-CHARS
                  LINEARIZE-COMMENT MAP-COMMENT-INDEX PARSE--COMMENT PARSE--COMMENT-WORD PARSE-STRING-INTO-WORDS
                  SELECT-SEGMENT-COMMENT SET-POINT-COMMENT SET-POINT-COMMENT-WORD SET-SELECTION-COMMENT
                  SET-SELECTION-COMMENT-WORD SIMPLE-STRING-OFFSET SIMPLE-STRING-SCAN START-COMMENT
                  STRINGIFY-COMMENT CREATE-COMMENT-WORD-NODE CREATE-COMMENT-WORD-NODES UNDO-COMMENT-CHANGE
                  UPGRADE-COMMENT)
           (IL:FUNCTIONS MAKE-COMMENT-STRING VERIFY-COMMENT)))

(IL:PUTPROPS IL:SEDIT-COMMENTS IL:FILETYPE :COMPILE-FILE)

(IL:PUTPROPS IL:SEDIT-COMMENTS IL:MAKEFILE-ENVIRONMENT (:READTABLE "XCL" :PACKAGE (DEFPACKAGE IL:SEDIT
                                                                                    (:USE IL:LISP IL:XCL))
                                                                                    ))

(IL:DECLARE\: IL:DOEVAL@COMPILE IL:DONTCOPY

(IL:LOCALVARS . T)
)

(IL:DECLARE\: IL:DONTCOPY IL:DOEVAL@COMPILE

(IL:FILESLOAD IL:SEDIT-DECLS)
)

(IL:DECLARE\: IL:EVAL@COMPILE

(IL:RPAQQ IL:LEVEL-1-COMMENT IL:\;)

(IL:RPAQQ IL:LEVEL-2-COMMENT IL:|;;|)

(IL:RPAQQ IL:LEVEL-3-COMMENT IL:|;;;|)

(IL:RPAQQ IL:LEVEL-4-COMMENT IL:|;;;;|)

(IL:RPAQQ IL:LEVEL-5-COMMENT IL:\|)

(IL:RPAQ IL:COMMENT-LEVEL-TABLE (LIST LEVEL-1-COMMENT 1 LEVEL-2-COMMENT 2 LEVEL-3-COMMENT 3 LEVEL-4-COMMENT 4
                                      LEVEL-5-COMMENT 5))

(IL:RPAQ IL:COMMENT-MARKERS (LIST LEVEL-1-COMMENT LEVEL-2-COMMENT LEVEL-3-COMMENT LEVEL-4-COMMENT
                                  LEVEL-5-COMMENT))

(IL:CONSTANTS (LEVEL-1-COMMENT 'IL:\;)
       (LEVEL-2-COMMENT 'IL:|;;|)
       (LEVEL-3-COMMENT 'IL:|;;;|)
       (LEVEL-4-COMMENT 'IL:|;;;;|)
       (LEVEL-5-COMMENT 'IL:\|)
       (COMMENT-LEVEL-TABLE (LIST LEVEL-1-COMMENT 1 LEVEL-2-COMMENT 2 LEVEL-3-COMMENT 3 LEVEL-4-COMMENT 4
                                  LEVEL-5-COMMENT 5))
       (COMMENT-MARKERS (LIST LEVEL-1-COMMENT LEVEL-2-COMMENT LEVEL-3-COMMENT LEVEL-4-COMMENT LEVEL-5-COMMENT)))
)

(IL:DEFINEQ

(BACKSPACE-COMMENT
  (IL:LAMBDA (NODE CONTEXT INDEX)                    ; Edited  7-Jul-87 09:50 by DCB
                                                     ; the BackSpace method for comments
```

```
    (COND
       ((NULL INDEX)                                          ; backspacing from the right boundary puts the caret immediately
                                                              ; after the last character
        (LET ((POINT (IL:FETCH CARET-POINT IL:OF CONTEXT)))
             (CLOSE-OPEN-NODE CONTEXT)
             (IL:REPLACE POINT-NODE IL:OF POINT IL:WITH NODE)
             (IL:REPLACE POINT-INDEX IL:OF POINT IL:WITH (IL:NCHARS (CADDR (IL:FETCH STRUCTURE IL:OF NODE))))
             (IL:REPLACE POINT-TYPE IL:OF POINT IL:WITH 'STRING)))
       ((EQ 0 INDEX)
        (COND
           ((IL:IGREATERP (IL:FETCH UNASSIGNED IL:OF NODE)
                  1)                                          ; backspacing over one of the semicolons
            (DEGRADE-COMMENT CONTEXT NODE))
           ((NULL (CDR (IL:FETCH SUB-NODES IL:OF NODE)))      ; backspacing from the front of an empty comment deletes it
            (DELETE-NODES (IL:FETCH SUPER-NODE IL:OF NODE)
                   CONTEXT NODE NIL (IL:FETCH CARET-POINT IL:OF CONTEXT)))))
       (T                                                     ; otherwise, delete the character to the left of the caret
          (DELETE-COMMENT NODE CONTEXT INDEX NIL (IL:FETCH CARET-POINT IL:OF CONTEXT))))
    (SET-SELECTION-NOWHERE (IL:FETCH SELECTION IL:OF CONTEXT))))
```

(CFV-COMMENT

```
  (IL:LAMBDA (NODE ENVIRONMENT CONTEXT FORMAT)                ; Edited 12-Feb-88 16:48 by raf
```

;;; compute the width estimates for a comment node

```
    (IL:REPLACE INLINE-WIDTH IL:OF NODE IL:WITH NIL)          ; dispatch on the comment level
    (LET ((WIDTH (IL:FETCH COMMENT-WIDTH IL:OF CONTEXT)))
         (IL:SELECTQ (IL:FETCH UNASSIGNED IL:OF NODE)
                (1 ;; here we know the comment width
                   (IL:REPLACE PREFERRED-WIDTH IL:OF NODE IL:WITH WIDTH))
                (2 ;; these affect the super-node's formatting.  We don't generally want double-semi comments to force us into miser mode, so
                   ;; guess small
                   (IL:REPLACE PREFERRED-WIDTH IL:OF NODE IL:WITH 30))
                ((3 4 5) ;; since these won't affect supernode's formattng, just guess small
                      (IL:REPLACE PREFERRED-WIDTH IL:OF NODE IL:WITH 30))
                (IL:SHOULDNT "unexpected value for comment level")))))
```

(CLOSE-NODE-COMMENT

```
  (IL:LAMBDA (CONTEXT NODE)                                   ; Edited 13-Apr-88 14:45 by woz
    (UNDO-BY UNDO-COMMENT-CHANGE NODE (CADDR (IL:FETCH STRUCTURE IL:OF NODE)))
    (RPLACA (CDDR (IL:|fetch| STRUCTURE IL:|of| NODE))
           (MAKE-COMMENT-STRING NODE))
    (IL:|replace| OPEN-NODE IL:|of| CONTEXT IL:|with| NIL)))
```

(COMMENT-LENGTH

```
  (IL:LAMBDA (NODE NUMBER-OF-SUBNODES)                        ; Edited 13-Apr-88 14:25 by woz
    (IL:|for| I IL:|from| 1 IL:|to| NUMBER-OF-SUBNODES IL:|as| SUBNODE IL:|in| (CDR (IL:|fetch| SUB-NODES IL:|of| NODE))
       IL:|sum| (IL:NCHARS (IL:|fetch| STRUCTURE IL:|of| SUBNODE)))))
```

(COMPUTE-COMMENT-COLUMN

```
  (IL:LAMBDA (CONTEXT WINDOW)                                 ; Edited  7-Jul-87 09:50 by DCB
    (LET ((ENVIRONMENT (IL:FETCH ENVIRONMENT IL:OF CONTEXT)))
         ;; set the context's comment column info based on the window.
         (IL:REPLACE COMMENT-WIDTH IL:OF CONTEXT IL:WITH (IL:IQUOTIENT (IL:ITIMES (IL:WINDOWPROP WINDOW
                                                                                  'IL:WIDTH)
                                                                                  (IL:FETCH COMMENT-WIDTH-PERCENT
                                                                                     IL:OF ENVIRONMENT))
                                                                                  100))
         (IL:REPLACE COMMENT-SEPARATION IL:OF CONTEXT IL:WITH (IL:FETCH INIT-COMMENT-SEPARATION IL:OF ENVIRONMENT)
                )))))
```

(COMPUTE-POINT-POSITION-COMMENT

```
  (IL:LAMBDA (POINT CONTEXT)                                  ; Edited 17-Nov-87 11:47 by DCB
```

;;; implements the ComputePointPosition method for a comment

```
    (LET ((NODE (IL:FETCH POINT-NODE IL:OF POINT))
          SUBNODE)
         (MAP-COMMENT-INDEX CONTEXT NODE (IL:FETCH POINT-INDEX IL:OF POINT))
         (IL:SETQ SUBNODE (CAR (IL:FETCH \\Y IL:OF CONTEXT)))
         (COND
            ((NULL SUBNODE)
             (IL:REPLACE POINT-X IL:OF POINT IL:WITH (IL:IPLUS (IL:FETCH START-X IL:OF NODE)
                                                               (IL:FETCH WIDTH IL:OF (CAR (IL:FETCH LINEAR-FORM
                                                                                             IL:OF NODE)))))
             (IL:REPLACE POINT-LINE IL:OF POINT IL:WITH (IL:FETCH FIRST-LINE IL:OF NODE)))
            (T (IL:REPLACE POINT-LINE IL:OF POINT IL:WITH (IL:FETCH FIRST-LINE IL:OF SUBNODE))
               (IL:REPLACE POINT-X IL:OF POINT IL:WITH (IL:IPLUS (IL:FETCH START-X IL:OF SUBNODE)
```

```
                                                    (SIMPLE-STRING-OFFSET (CAR (IL:FETCH LINEAR-FORM
                                                                                 IL:OF SUBNODE))
                                                                          (IL:FETCH \\X IL:OF CONTEXT)))))))))
```

(COMPUTE-SELECTION-POSITION-COMMENT

```
(IL:LAMBDA (SELECTION CONTEXT)                                  ; Edited 17-Nov-87 11:48 by DCB
```

;;; implements the ComputeSelectionPosition method for a comment

```
    (LET ((NODE (IL:FETCH SELECT-NODE IL:OF SELECTION))
          (START (IL:FETCH SELECT-START IL:OF SELECTION)))
         (MAP-COMMENT-INDEX CONTEXT NODE START (OR (IL:FETCH SELECT-END IL:OF SELECTION)
                                                   START))
         (LET ((START-SUBNODE (CAR (IL:FETCH \\Y IL:OF CONTEXT)))
               (END-SUBNODE (CAR (IL:FETCH \\T IL:OF CONTEXT))))
              (IL:REPLACE SELECT-START-LINE IL:OF SELECTION IL:WITH (IL:FETCH FIRST-LINE IL:OF START-SUBNODE))
              (IL:REPLACE SELECT-START-X IL:OF SELECTION IL:WITH (IL:IPLUS (IL:FETCH START-X IL:OF START-SUBNODE)
                                                                           (SIMPLE-STRING-OFFSET (CAR (IL:FETCH
                                                                                                       LINEAR-FORM
                                                                                                      IL:OF
                                                                                                      START-SUBNODE
                                                                                                       ))
                                                                                                 (IL:SUB1 (IL:FETCH \\X IL:OF
                                                                                                                    CONTEXT
                                                                                                 )))))
              (IL:REPLACE SELECT-END-LINE IL:OF SELECTION IL:WITH (IL:FETCH FIRST-LINE IL:OF END-SUBNODE))
              (IL:REPLACE SELECT-END-X IL:OF SELECTION IL:WITH (IL:IPLUS (IL:FETCH START-X IL:OF END-SUBNODE)
                                                                         (SIMPLE-STRING-OFFSET (CAR (IL:FETCH
                                                                                                     LINEAR-FORM
                                                                                                    IL:OF
                                                                                                     END-SUBNODE
                                                                                                      ))
                                                                                               (IL:FETCH \\Z IL:OF CONTEXT))))))))
```

(COPY-SELECTION-COMMENT

```
(IL:LAMBDA (SELECTION CONTEXT DESTINATION POINT DELETE?)        ; Edited 23-Feb-88 11:37 by raf
```

;;; method for shift selecting a comment anywhere.

```
    (LET ((NODE (IL:FETCH SELECT-NODE IL:OF SELECTION))
          (COMMENT (CADDR (IL:FETCH STRUCTURE IL:OF (IL:FETCH SELECT-NODE IL:OF SELECTION))))
          (START (IL:FETCH SELECT-START IL:OF SELECTION))
          (END (IL:FETCH SELECT-END IL:OF SELECTION))
          (PROMPTWINDOW (GET-PROMPT-WINDOW (OR DESTINATION CONTEXT)))
          INSERT)
         (COND
            ((AND START (OR (IL:NEQ (OR END (IL:SETQ END START))
                                    (IL:SUB1 (IL:NCHARS COMMENT)))
                            (IL:NEQ START 0)))                  ; some subset of the comment has been selected
             (IL:SETQ COMMENT (IL:SUBSTRING COMMENT START END)))
            (T (IL:SETQ COMMENT (STRINGIFY-COMMENT NODE (IL:FETCH ENVIRONMENT IL:OF CONTEXT)))))
         (WHEN DELETE? (DELETE-NODES NODE CONTEXT START END))
         (COND
            ((NULL DESTINATION)                                 ; it's going to a foreign sink;  bksysbuf it
             (IL:BKSYSBUF COMMENT (AND (EQ (IL:FETCH NODE-TYPE IL:OF NODE)
                                           TYPE-STRING)
                                       (NULL START))))
            ((EQ (IL:FETCH POINT-TYPE IL:OF POINT)
                 'STRING)                                       ; comments insert as whole structures
             (INSERT POINT DESTINATION COMMENT))
            (T (WHEN (EQ (IL:FETCH POINT-TYPE IL:OF POINT)
                         'ATOM)                                 ; first make a structure point
                  (INSERT POINT DESTINATION NIL))
               (COND
                  ((NOT START)                                  ; insert whole node
                   (INSERT POINT DESTINATION (COPY-NODE NODE DESTINATION)))
                  (T (IL:SETQ INSERT (IL:BIND (STREAM IL:_ (IL:OPENSTRINGSTREAM COMMENT))
                                              OBJ IL:WHILE (IL:SETQ OBJ (IL:NLSETQ (IL:READ STREAM)))
                                        IL:COLLECT (PARSE-NEW (CAR OBJ)
                                                              DESTINATION)))
                     (IF INSERT
                         (INSERT POINT DESTINATION INSERT)
                         (IL:|printout| PROMPTWINDOW T "Selection not a valid structure."))))))))
```

(COPY-STRUCTURE-COMMENT

```
(IL:LAMBDA (NODE)                                               ; Edited 13-Apr-88 14:44 by woz
  (IL:|replace| STRUCTURE IL:|of| NODE IL:|with| (LIST 'IL:* (CADR (IL:|fetch| STRUCTURE IL:|of| NODE))
                                                       (MAKE-COMMENT-STRING NODE)))))
```

(COPY-STRUCTURE-COMMENT-WORD

```
(IL:LAMBDA (NODE)                                               ; Edited 13-Apr-88 14:28 by woz
```

;;; the structure field of the new comment.word isn't completely built here, since it's supposed to be a list of all the words in the comment starting with this
;;; one. instead, we build one element lists for each comment.word, and copy.structure.comment links them all together

```
    (LET ((NEW-STRING (COPY-SEQ (IL:|fetch| STRUCTURE IL:|of| NODE))))
         (IL:|replace| STRUCTURE IL:|of| NODE IL:|with| NEW-STRING)
         (RPLACA (IL:|fetch| LINEAR-FORM IL:|of| NODE)
                (IL:|create| STRING-ITEM IL:|using| (CAR (IL:|fetch| LINEAR-FORM IL:|of| NODE))
                                          STRING IL:_ NEW-STRING)))))
```

(CREATE-NEW-COMMENT

```
  (IL:LAMBDA (CONTEXT)                                          ; Edited  6-Apr-88 16:35 by woz
    (LET* ((WIDTH (IL:|fetch| COMMENT-WIDTH IL:|of| CONTEXT))
           (COMMENT (IL:|create| EDIT-NODE
                          NODE-TYPE IL:_ TYPE-COMMENT
                          STRUCTURE IL:_ (LIST 'IL:* 'IL:\; "")
                          DEPTH IL:_ 0
                          INLINE-WIDTH IL:_ NIL
                          PREFERRED-WIDTH IL:_ WIDTH
                          UNASSIGNED IL:_ 1
                          SUB-NODES IL:_ (LIST 0))))
          (IL:|replace| LINEAR-FORM IL:|of| COMMENT IL:|with| (CREATE-WEAK-LINK COMMENT))
          COMMENT)))
```

(DEGRADE-COMMENT

```
  (IL:LAMBDA (CONTEXT NODE)                                     ; Edited  7-Jul-87 09:53 by DCB
    (RPLACA (CDR (IL:|FETCH| STRUCTURE IL:|OF| NODE))
           (CAR (IL:NTH COMMENT-MARKERS (IL:|ADD| (IL:|FETCH| UNASSIGNED IL:|OF| NODE)
                                                  -1))))
    (NOTE-CHANGE NODE CONTEXT)
    (WHEN (IL:|FETCH| SUPER-NODE IL:|OF| (IL:|FETCH| SUPER-NODE IL:|OF| NODE))
                                                                ; this node has a supernode that is not the root
        (NOTE-CHANGE (IL:|FETCH| SUPER-NODE IL:|OF| NODE)
                CONTEXT))
    (UNDO-BY UPGRADE-COMMENT NODE)))
```

(DELETE-COMMENT

```
  (IL:LAMBDA (NODE CONTEXT START END SET-POINT?)                ; Edited 27-Apr-88 11:14 by woz
```

;;; the Delete method for comments

```
    (WHEN (IL:NEQ (IL:|fetch| OPEN-NODE IL:|of| CONTEXT)
                  NODE)
        (CLOSE-OPEN-NODE CONTEXT)
        (IL:|replace| OPEN-NODE IL:|of| CONTEXT IL:|with| NODE))
    (IL:|replace| OPEN-NODE-CHANGED? IL:|of| CONTEXT IL:|with| T)
    (WHEN SET-POINT?
        (IL:|replace| POINT-NODE IL:|of| SET-POINT? IL:|with| NODE)
        (IL:|replace| POINT-INDEX IL:|of| SET-POINT? IL:|with| (IL:SUB1 START))
        (IL:|replace| POINT-TYPE IL:|of| SET-POINT? IL:|with| 'STRING))
    (MAP-COMMENT-INDEX CONTEXT NODE START (OR END START))
    (PROG* ((START-INDEX (IL:|fetch| \\X IL:|of| CONTEXT))
            (START-NODE (CAR (IL:|fetch| \\Y IL:|of| CONTEXT)))
            (END-INDEX (IL:|fetch| \\Z IL:|of| CONTEXT))
            (END-NODE (CAR (IL:|fetch| \\T IL:|of| CONTEXT)))
            (NUMBER-OF-SUBNODES (CAR (IL:|fetch| SUB-NODES IL:|of| NODE)))
            NODE-INDEX STRING LENGTH NEW-WIDTH)
           (WHEN (EQ START-NODE END-NODE)
               (IL:SETQ STRING (IL:|fetch| STRING IL:|of| (CAR (IL:|fetch| LINEAR-FORM IL:|of| START-NODE))))
               (IL:SETQ LENGTH (IL:NCHARS STRING))
               (IL:SETQ NODE-INDEX (IL:|fetch| SUB-NODE-INDEX IL:|of| START-NODE))
               (WHEN (NOT (OR (AND (EQ START-INDEX 1)
                                   (IL:NEQ NODE-INDEX 1)
                                   (OR (EQ END-INDEX LENGTH)
                                       (EQ (IL:NTHCHARCODE STRING (IL:ADD1 END-INDEX))
                                           (IL:CHARCODE IL:SP))))
                              (AND (EQ END-INDEX LENGTH)
                                   (IL:NEQ NODE-INDEX NUMBER-OF-SUBNODES)
                                   (OR (EQ START-INDEX 1)
                                       (IL:NEQ (IL:NTHCHARCODE STRING (IL:SUB1 START-INDEX))
                                               (IL:CHARCODE IL:SP))))))
                   ;; we're not going to merge -- fast case
                   (COND
                      ((AND (EQ START-INDEX 1)
                            (EQ END-INDEX LENGTH))              ; everything deleted!
                       (IL:|replace| OPEN-NODE IL:|of| CONTEXT IL:|with| NIL)
                       (IL:|replace| OPEN-NODE-CHANGED? IL:|of| CONTEXT IL:|with| NIL)
                       (RPLACA (CDDR (IL:|fetch| STRUCTURE IL:|of| NODE))
                               (IL:CONCAT ""))
                       (IL:|replace| SUB-NODES IL:|of| NODE IL:|with| (LIST 0))
                       (NOTE-CHANGE NODE CONTEXT))
                      (T (IL:SETQ NEW-WIDTH (IL:IDIFFERENCE (IL:|fetch| INLINE-WIDTH IL:|of| START-NODE)
                                                            (STRINGWIDTH (IL:SUBSTRING STRING START-INDEX END-INDEX)
                                                                         (IL:|fetch| FONT IL:|of| (CAR (IL:|fetch| LINEAR-FORM
                                                                                                                  IL:|of| START-NODE))))))
                         (IL:SETQ STRING (IL:CONCAT (OR (IL:SUBSTRING STRING 1 (IL:SUB1 START-INDEX))
```

```
                                                    "")
                                                (OR (IL:SUBSTRING STRING (IL:ADD1 END-INDEX))
                                                    "")))
                                (IL:|replace| STRING IL:|of| (CAR (IL:|fetch| LINEAR-FORM IL:|of| START-NODE)) IL:|with| STRING)
                                (IL:|replace| STRING IL:|of| START-NODE IL:|with| STRING)
                                (ADJUST-WIDTH START-NODE CONTEXT NEW-WIDTH)))
                        (LET ((CARET (IL:|fetch| CARET-POINT IL:|of| CONTEXT)))
                             (WHEN (AND (IL:NEQ CARET SET-POINT?)
                                        (EQ (IL:|fetch| POINT-NODE IL:|of| CARET)
                                            NODE)
                                        (IL:IGEQ (IL:|fetch| POINT-INDEX IL:|of| CARET)
                                                 START))
                                   ;; if the caret was within or after replaced characters, it will need to be fixed up
                                   (IL:|replace| POINT-INDEX IL:|of| CARET IL:|with| (IL:IDIFFERENCE (IL:|fetch| POINT-INDEX
                                                                                                      IL:|of| CARET)
                                                                                     (IL:IDIFFERENCE (IL:ADD1 END-INDEX)
                                                                                            START-INDEX)))))
                        (RETURN)))
                (IL:SETQ LENGTH (IL:NCHARS (CADDR (IL:|fetch| STRUCTURE IL:|of| NODE))))
                                                                                ; save old length
                (IL:SETQ STRING (IL:CONCAT (OR (IL:SUBSTRING (IL:|fetch| STRUCTURE IL:|of| START-NODE)
                                                             1
                                                             (IL:SUB1 START-INDEX))
                                               "")
                                           (OR (IL:SUBSTRING (IL:|fetch| STRUCTURE IL:|of| END-NODE)
                                                             (IL:ADD1 END-INDEX))
                                               "")))
                (IL:|for| SUBNODE-INDEX IL:|from| (IL:|fetch| SUB-NODE-INDEX IL:|of| START-NODE) IL:|bind| NODES REST-NODES
                   IL:|first| (IL:SETQ NODES (IL:NTH (IL:|fetch| SUB-NODES IL:|of| NODE)
                                                     SUBNODE-INDEX))
                        (IL:SETQ REST-NODES (CDR (IL:|fetch| \\T IL:|of| CONTEXT)))
                        (RPLACD NODES REST-NODES)
                   IL:|while| REST-NODES IL:|do| (IL:|replace| SUB-NODE-INDEX IL:|of| (CAR REST-NODES) IL:|with| SUBNODE-INDEX)
                                                 (IL:SETQ REST-NODES (CDR REST-NODES))
                   IL:|finally| (RPLACA (IL:|fetch| SUB-NODES IL:|of| NODE)
                                        (IL:SUB1 SUBNODE-INDEX))
                        (WHEN (IL:IGREATERP (IL:NCHARS STRING)
                                            0)
                              (INSERT-COMMENT-CHARS CONTEXT NODE (AND (IL:NEQ (IL:|fetch| SUB-NODE-INDEX IL:|of| START-NODE
                                                                                                                    )
                                                                                                              1)
                                                                                                       NODES)
                                                                      NIL STRING)))
                (NOTE-CHANGE NODE CONTEXT)
                (LET ((CARET (IL:|fetch| CARET-POINT IL:|of| CONTEXT)))
                     (WHEN (AND (IL:NEQ CARET SET-POINT?)
                                (EQ (IL:|fetch| POINT-NODE IL:|of| CARET)
                                    NODE)
                                (IL:IGEQ (IL:|fetch| POINT-INDEX IL:|of| CARET)
                                         START))
                           ;; if the caret was within or after replaced characters, it will need to be fixed up
                           (IL:|replace| POINT-INDEX IL:|of| CARET IL:|with| (IL:IDIFFERENCE (IL:|fetch| POINT-INDEX IL:|of| CARET)
                                                                                             (IL:IDIFFERENCE (IL:ADD1 (OR END START))
                                                                                                    START))))))
   T))
```

(INITIALIZE-COMMENTS

```
  (IL:LAMBDA NIL                                                    ; Edited 7-Jul-87 09:54 by DCB
    (IL:SETQ TYPES (LIST* (IL:SETQ TYPE-COMMENT (IL:CREATE EDIT-NODE-TYPE
                                                           NAME IL:_ 'COMMENT
                                                           ASSIGN-FORMAT IL:_ 'IL:NILL
                                                           COMPUTE-FORMAT-VALUES IL:_ 'CFV-COMMENT
                                                           LINEARIZE IL:_ 'LINEARIZE-COMMENT
                                                           SET-POINT IL:_ 'SET-POINT-COMMENT
                                                           SET-SELECTION IL:_ 'SET-SELECTION-COMMENT
                                                           GROW-SELECTION IL:_ 'GROW-SELECTION-LITATOM
                                                           SELECT-SEGMENT IL:_ 'SELECT-SEGMENT-COMMENT
                                                           COMPUTE-POINT-POSITION IL:_
                                                           'COMPUTE-POINT-POSITION-COMMENT
                                                           COMPUTE-SELECTION-POSITION IL:_
                                                           'COMPUTE-SELECTION-POSITION-COMMENT
                                                           INSERT IL:_ 'INSERT-COMMENT
                                                           DELETE IL:_ 'DELETE-COMMENT
                                                           COPY-STRUCTURE IL:_ 'COPY-STRUCTURE-COMMENT
                                                           COPY-SELECTION IL:_ 'COPY-SELECTION-COMMENT
                                                           STRINGIFY IL:_ 'STRINGIFY-COMMENT
                                                           BACK-SPACE IL:_ 'BACKSPACE-COMMENT
                                                           CLOSE-NODE IL:_ 'CLOSE-NODE-COMMENT))
                          (IL:SETQ TYPE-COMMENT-WORD (IL:CREATE EDIT-NODE-TYPE
                                                                NAME IL:_ 'COMMENT-WORD
                                                                ASSIGN-FORMAT IL:_ 'IL:NILL
                                                                COMPUTE-FORMAT-VALUES IL:_ 'IL:NILL
                                                                SET-POINT IL:_ 'SET-POINT-COMMENT-WORD
```

```
                                                    SET-SELECTION IL:_ 'SET-SELECTION-COMMENT-WORD
                                                    COPY-STRUCTURE IL:_ 'COPY-STRUCTURE-COMMENT-WORD))
                        TYPES))))
```

(INSERT-COMMENT

```
(IL:LAMBDA (NODE CONTEXT WHERE CHARS POINT)                    ; Edited 17-Jul-87 09:59 by DCB
```

;;; the Insert method for comments

```
    (LET (START)
         (COND
            ((IL:TYPE? EDIT-SELECTION WHERE)
             (IL:SETQ START (IL:SUB1 (IL:FETCH SELECT-START IL:OF WHERE)))
             (DELETE-COMMENT NODE CONTEXT (IL:ADD1 START)
                    (OR (IL:FETCH SELECT-END IL:OF WHERE)
                        (IL:ADD1 START))))
            (T (IL:SETQ START (IL:FETCH POINT-INDEX IL:OF WHERE))))
         (COND
            (CHARS (MAP-COMMENT-INDEX CONTEXT NODE START)
                   (WHEN (IL:NEQ (IL:FETCH OPEN-NODE IL:OF CONTEXT)
                                 NODE)
                       (CLOSE-OPEN-NODE CONTEXT)
                       (IL:REPLACE OPEN-NODE IL:OF CONTEXT IL:WITH NODE))
                   (IL:REPLACE OPEN-NODE-CHANGED? IL:OF CONTEXT IL:WITH T)
                   (INSERT-COMMENT-CHARS CONTEXT NODE (IL:FETCH \\Y IL:OF CONTEXT)
                          (IL:FETCH \\X IL:OF CONTEXT)
                          CHARS)
                   (NOTE-CHANGE NODE CONTEXT)
                   (WHEN POINT
                       (IL:REPLACE POINT-NODE IL:OF POINT IL:WITH NODE)
                       (IL:REPLACE POINT-INDEX IL:OF POINT IL:WITH (IL:IPLUS START (IL:NCHARS CHARS)))))
            (T (SPLIT-COMMENT NODE POINT CONTEXT START))))))
```

(SPLIT-COMMENT

```
(IL:LAMBDA (NODE POINT CONTEXT START)                          ; Edited  7-Jul-87 09:54 by DCB
  (CLOSE-OPEN-NODE CONTEXT)
  (LET* ((COMMENT (CADDR (IL:FETCH STRUCTURE IL:OF NODE)))
         (LENGTH (IL:NCHARS COMMENT))
         (SPLIT-STRING (IL:SUBSTRING COMMENT (IL:ADD1 START)
                                     LENGTH)))
        (SET-POINT POINT CONTEXT (IL:FETCH SUPER-NODE IL:OF NODE)
               (IL:FETCH SUB-NODE-INDEX IL:OF NODE)
               T NODE 'STRUCTURE)
        (WHEN (IL:NEQ START LENGTH)                            ; split in middle of comment.
            (DELETE-NODES NODE CONTEXT (IL:ADD1 START)
                  LENGTH NIL COMMENT)
            (INSERT POINT CONTEXT (PARSE-NEW (LIST 'IL:* (CAR (IL:NTH COMMENT-MARKERS (IL:FETCH UNASSIGNED
                                                                                       IL:OF NODE)))
                                                   SPLIT-STRING)
                                             CONTEXT))
            (SET-POINT POINT CONTEXT (IL:FETCH SUPER-NODE IL:OF NODE)
                   (IL:FETCH SUB-NODE-INDEX IL:OF NODE)
                   T NODE 'STRUCTURE)))))
```

(INSERT-COMMENT-CHARS

```
(IL:LAMBDA (CONTEXT NODE SUBNODES INDEX CHARS)                 ; Edited 13-Apr-88 16:55 by woz
```

;;; what a hack. ugh blech.

```
    (LET ((LENGTH (IL:NCHARS CHARS))
          (SUBNODE (CAR SUBNODES))
          (FONT (IL:|fetch| COMMENT-FONT IL:|of| (IL:|fetch| ENVIRONMENT IL:|of| CONTEXT)))
          STRING STRING-LENGTH)
         (WHEN SUBNODE
             (IL:SETQ STRING (IL:|fetch| STRUCTURE IL:|of| SUBNODE))
             (IL:SETQ STRING-LENGTH (IL:NCHARS STRING))
             (WHEN (EQ INDEX STRING-LENGTH)
                   (IL:SETQ INDEX NIL)))
         (COND
            ((AND (EQ LENGTH 1)
                  SUBNODE
                  (IF (EQ (IL:CHCON1 CHARS)
                          (IL:CHARCODE IL:SP))
                      (OR (NULL INDEX)
                          (EQ (IL:NTHCHARCODE STRING (IL:ADD1 INDEX))
                              (IL:CHARCODE IL:SP)))
                      (OR (EQ INDEX 0)
                          (IL:NEQ (IL:NTHCHARCODE STRING (OR INDEX STRING-LENGTH))
                                  (IL:CHARCODE IL:SP)))))
             ;; fast case
             (IL:SETQ CHARS (IL:MKSTRING CHARS))
             (IL:SETQ STRING (IL:CONCAT (OR (IL:SUBSTRING STRING 1 INDEX)
                                            "")
```

```
                                CHARS
                                (OR (AND INDEX (IL:SUBSTRING STRING (IL:ADD1 INDEX)))
                                    ""))))
          (IL:|replace| STRING IL:|of| (CAR (IL:|fetch| LINEAR-FORM IL:|of| SUBNODE)) IL:|with| STRING)
          (IL:|replace| STRUCTURE IL:|of| SUBNODE IL:|with| STRING)
          (ADJUST-WIDTH SUBNODE CONTEXT (IL:IPLUS (IL:|fetch| INLINE-WIDTH IL:|of| SUBNODE)
                                                  (STRINGWIDTH CHARS FONT))))
        (T (COND
             ((EQ INDEX 0)
              (IL:SETQ SUBNODES (AND (IL:NEQ (IL:|fetch| SUB-NODE-INDEX IL:|of| SUBNODE)
                                             1)
                                     (IL:NTH (IL:|fetch| SUB-NODES IL:|of| NODE)
                                             (IL:|fetch| SUB-NODE-INDEX IL:|of| SUBNODE))))
              (IL:SETQ SUBNODE (CAR SUBNODES)))
             (INDEX (LET* ((NEW-STRING (IL:SUBSTRING STRING (IL:ADD1 INDEX)))
                           (NEW-SUBNODE (CREATE-SIMPLE-NODE NEW-STRING (IL:|fetch| ENVIRONMENT IL:|of| CONTEXT)
                                                            TYPE-COMMENT-WORD NEW-STRING NIL FONT)))
                          (ADJUST-WIDTH SUBNODE CONTEXT (IL:IDIFFERENCE (IL:|fetch| INLINE-WIDTH IL:|of| SUBNODE)
                                                                        (IL:|fetch| INLINE-WIDTH IL:|of| NEW-SUBNODE)))
                          (RPLACD SUBNODES (CONS NEW-SUBNODE (CDR SUBNODES)))
                          (IL:SETQ NEW-STRING (IL:SUBSTRING STRING 1 INDEX))
                          (IL:|replace| STRING IL:|of| (CAR (IL:|fetch| LINEAR-FORM IL:|of| SUBNODE)) IL:|with|
                                                                                                      NEW-STRING
                                        )
                          (IL:|replace| STRUCTURE IL:|of| SUBNODE IL:|with| NEW-STRING))))
           (LET ((WORDS (CREATE-COMMENT-WORD-NODES CHARS (IF SUBNODES
                                                             (CDR SUBNODES)
                                                             (CDR (IL:|fetch| SUB-NODES IL:|of| NODE)))
                                                   (IL:|fetch| ENVIRONMENT IL:|of| CONTEXT))))
             (IF SUBNODES
                 (RPLACD SUBNODES WORDS)
                 (RPLACD (IL:|fetch| SUB-NODES IL:|of| NODE)
                         WORDS)))
           (IL:|for| IL:|old| SUBNODES IL:|on| (CDR (IL:|fetch| SUB-NODES IL:|of| NODE))
              IL:|bind| (N IL:_ 0)
                   NEXT-SUBNODE STRING (DEPTH IL:_ (IL:ADD1 (IL:|fetch| DEPTH IL:|of| NODE)))
              IL:|do| (IL:SETQ N (IL:ADD1 N))
                   (IL:SETQ SUBNODE (CAR SUBNODES))
                   (IL:|replace| SUB-NODE-INDEX IL:|of| SUBNODE IL:|with| N)
                   (IL:|replace| SUPER-NODE IL:|of| SUBNODE IL:|with| NODE)
                   (IL:|replace| DEPTH IL:|of| SUBNODE IL:|with| DEPTH)
                   (IL:SETQ STRING (IL:|fetch| STRUCTURE IL:|of| SUBNODE))
                   (IL:|while| (AND (IL:SETQ NEXT-SUBNODE (CADR SUBNODES))
                                    (OR (IL:NEQ (IL:NTHCHARCODE STRING (IL:NCHARS STRING))
                                                (IL:CHARCODE IL:SP))
                                        (EQ (IL:CHCON1 (CAR (IL:|fetch| STRUCTURE IL:|of| NEXT-SUBNODE)))
                                            (IL:CHARCODE IL:SP))))
                      IL:|do| (IL:SETQ STRING (IL:CONCAT STRING (IL:|fetch| STRUCTURE IL:|of| NEXT-SUBNODE)))
                              (IL:|replace| STRUCTURE IL:|of| SUBNODE IL:|with| STRING)
                              (IL:|replace| STRING IL:|of| (CAR (IL:|fetch| LINEAR-FORM IL:|of| SUBNODE)) IL:|with| STRING)
                              (ADJUST-WIDTH SUBNODE CONTEXT (IL:IPLUS (IL:|fetch| INLINE-WIDTH IL:|of| SUBNODE)
                                                                      (IL:|fetch| INLINE-WIDTH IL:|of| NEXT-SUBNODE)))
                              (RPLACD SUBNODES (CDDR SUBNODES)))
              IL:|finally| (RPLACA (IL:|fetch| SUB-NODES IL:|of| NODE)
                                   N)))))))
```

(**LINEARIZE-COMMENT**

```
(IL:LAMBDA (NODE CONTEXT INDEX)                                    ; Edited 23-Feb-88 11:18 by raf
  (LET* ((LEVEL (IL:|fetch| UNASSIGNED IL:|of| NODE))
         (ENVIRONMENT (IL:|fetch| ENVIRONMENT IL:|of| CONTEXT))
         (PREFIX (IL:LISTGET (IL:|fetch| COMMENT-STRING IL:|of| ENVIRONMENT)
                             LEVEL)))
        (IL:|bind| (IL:FIRST IL:_ T) IL:|for| SUBNODE IL:|in| (COND
                                                                 (INDEX (CDDR (IL:NTH (IL:|fetch| SUB-NODES IL:|of| NODE)
                                                                                      INDEX)))
                                                                 (T ;; we're at the beginning, so display the prefix
                                                                    (OUTPUT-CONSTANT-STRING CONTEXT PREFIX)
                                                                    (CDR (IL:|fetch| SUB-NODES IL:|of| NODE))))
          IL:|do| (COND
                    ((OR IL:FIRST (IL:ILEQ (IL:IPLUS (IL:|fetch| CURRENT-X IL:|of| CONTEXT)
                                                      (IL:|fetch| INLINE-WIDTH IL:|of| SUBNODE))
                                           (IL:|fetch| RIGHT-MARGIN IL:|of| NODE)))
                     (LINEARIZE SUBNODE CONTEXT))
                    (T (OUTPUT-CR CONTEXT (IL:|fetch| START-X IL:|of| NODE))
                       (UNLESS (EQ 5 LEVEL)
                               (OUTPUT-CONSTANT-STRING CONTEXT PREFIX))
                       (LINEARIZE SUBNODE CONTEXT)))
                  (IL:SETQ IL:FIRST NIL))
        (WHEN (EQ 5 LEVEL)
            (OUTPUT-CONSTANT-STRING CONTEXT (IL:LISTGET (IL:FETCH COMMENT-STRING IL:OF ENVIRONMENT)
                                                        6))))))
```

(**MAP-COMMENT-INDEX**

```
(IL:LAMBDA (CONTEXT NODE START END)                                ; Edited 13-Apr-88 14:26 by woz
```

```
    (IL:|bind| LENGTH SUBNODE (INDEX IL:_ START)
          (OPEN-NODE IL:_ (IL:|fetch| OPEN-NODE IL:|of| CONTEXT)) IL:|for| SUBNODES
       IL:|on| (CDR (IL:|fetch| SUB-NODES IL:|of| NODE))
       IL:|do| (IL:SETQ SUBNODE (CAR SUBNODES))
              (IL:SETQ LENGTH (IF (EQ SUBNODE OPEN-NODE)
                                  (IL:|fetch| REAL-LENGTH IL:|of| (IL:|fetch| STRING IL:|of| (CAR (IL:|fetch| LINEAR-FORM
                                                                                                   IL:|of| SUBNODE))))
                                  (IL:NCHARS (IL:|fetch| STRUCTURE IL:|of| SUBNODE))))
              (COND
                 ((IL:IGREATERP INDEX LENGTH)
                  (IL:SETQ INDEX (IL:IDIFFERENCE INDEX LENGTH)))
                 (T (WHEN START
                         (IL:|replace| \\X IL:|of| CONTEXT IL:|with| INDEX)
                         (IL:|replace| \\Y IL:|of| CONTEXT IL:|with| SUBNODES)
                         (WHEN (NULL END)
                               (RETURN))
                         (IL:SETQ INDEX (IL:IPLUS INDEX (IL:IDIFFERENCE END START)))
                         (WHEN (IL:IGREATERP INDEX LENGTH)
                             (IL:SETQ INDEX (IL:IDIFFERENCE INDEX LENGTH))
                             (IL:SETQ START NIL)
                             (IL:SETQ END NIL)
                             (GO IL:$$ITERATE)))
                    (IL:|replace| \\Z IL:|of| CONTEXT IL:|with| INDEX)
                    (IL:|replace| \\T IL:|of| CONTEXT IL:|with| SUBNODES)
                    (RETURN)))
       IL:|finally| (IL:|replace| \\X IL:|of| CONTEXT IL:|with| NIL)
              (IL:|replace| \\Y IL:|of| CONTEXT IL:|with| NIL)
              (IL:|replace| \\Z IL:|of| CONTEXT IL:|with| NIL)
              (IL:|replace| \\T IL:|of| CONTEXT IL:|with| NIL))))
```

(PARSE--COMMENT

```
  (IL:LAMBDA (STRUCTURE CONTEXT)                                    ; Edited 27-Apr-88 11:12 by woz
```

;;; try to parse this list as a common lisp comment. the second element should be one or more semicolons, and the rest of the list should be a string

```
    (LET (COMMENT-WORDS (LEVEL (AND (CDR STRUCTURE)
                                    (IL:LISTGET COMMENT-LEVEL-TABLE (CADR STRUCTURE)))))
         (WHEN (AND LEVEL (CDDR STRUCTURE)
                    (NULL (CDDDR STRUCTURE))
                    (IL:STRINGP (CADDR STRUCTURE))
                    (OR (NULL (IL:|fetch| CURRENT-NODE IL:|of| CONTEXT))
                        (IL:FMEMB (IL:|fetch| NAME IL:|of| (IL:|fetch| NODE-TYPE IL:|of| (IL:|fetch| CURRENT-NODE IL:|of| CONTEXT
                                                                                                                    )))
                                  '(FORM CLISP LAMBDA LIST))))
              (BUILD-NODE STRUCTURE CONTEXT TYPE-COMMENT T)
              (COND
                 ((NOT (IL:|fetch| \\X IL:|of| CONTEXT))
                  ;; if we're here for the first time then parse afresh.
                  (IL:SETQ COMMENT-WORDS (PARSE-STRING-INTO-WORDS (CADDR STRUCTURE)))
                  (IL:|for| WORD IL:|in| COMMENT-WORDS IL:|do| (PARSE WORD CONTEXT (IL:FUNCTION PARSE--COMMENT-WORD)))
                  (IL:|replace| UNASSIGNED IL:|of| (IL:|fetch| CURRENT-NODE IL:|of| CONTEXT) IL:|with| LEVEL))
                 ((AND NIL (NOT (VERIFY-COMMENT (IL:|fetch| CURRENT-NODE IL:|of| CONTEXT))))
                  ;; the comment changed from underneath us.  trash the subnodes and reparse.
                  ;; couldn't get this to work.  not absolutely at this point, so leave the case out.

                  )
                 (T ;; flag that everything matched.
                    (IL:|replace| \\X IL:|of| CONTEXT IL:|with| NIL)))
              T))))
```

(PARSE--COMMENT-WORD

```
  (IL:LAMBDA (STRUCTURE CONTEXT)                                    ; Edited  7-Jul-87 11:12 by DCB
```

;;; parse a comment word. different from string in that it does not use PRIN2 (does not print quotes round itself) and it uses a different font.

```
    (BUILD-PRELINEARIZED-NODE STRUCTURE CONTEXT TYPE-COMMENT-WORD STRUCTURE NIL (IL:FETCH COMMENT-FONT
                                                                                  IL:OF (IL:FETCH ENVIRONMENT
                                                                                         IL:OF CONTEXT)))))
```

(PARSE-STRING-INTO-WORDS

```
  (IL:LAMBDA (CHARS)                                                ; Edited  7-Jul-87 11:12 by DCB
    (IL:BIND (END IL:_ (IL:NCHARS CHARS))
          OK? RESULT I IL:FIRST (IL:SETQ I END) IL:WHILE (IL:NEQ I 0)
       IL:DO (COND
                ((IL:NEQ (IL:NTHCHARCODE CHARS I)
                         (IL:CHARCODE IL:SP))
                 (IL:SETQ OK? T))
                (OK? (IL:SETQ RESULT (CONS (IL:SUBSTRING CHARS (IL:ADD1 I)
                                                         END)
                                           RESULT))
                     (IL:SETQ END I)
```

```
                    (IL:SETQ OK? NIL)))
              (IL:SETQ I (IL:SUB1 I))
        IL:FINALLY (RETURN (AND (IL:NEQ END 0)
                                (CONS (IL:SUBSTRING CHARS 1 END)
                                      RESULT))))))
```

(SELECT-SEGMENT-COMMENT

```
(IL:LAMBDA (SELECTION CONTEXT NODE SUBNODE INDEX SUB-OFFSET SUB-ITEM)
                                                        ; Edited 17-Nov-87 11:54 by DCB
```

;;; the SelectSegment method for comments

```
    (LET ((START (IL:FETCH SELECT-START IL:OF SELECTION))
          NEW)
         (WHEN (AND START SUBNODE)
             (IL:SETQ NEW (IL:IPLUS (COMMENT-LENGTH NODE (IL:SUB1 (IL:FETCH SUB-NODE-INDEX IL:OF SUBNODE)))
                                    (SIMPLE-STRING-SCAN SUB-ITEM SUB-OFFSET)))
             (IL:REPLACE SELECT-END IL:OF SELECTION IL:WITH (IL:IMAX NEW (OR (IL:FETCH SELECT-END IL:OF SELECTION)
                                                                            START)))
             (WHEN (IL:ILESSP NEW START)
                   (IL:REPLACE SELECT-START IL:OF SELECTION IL:WITH NEW))
             (COMPUTE-SELECTION-POSITION-COMMENT SELECTION CONTEXT)))))
```

(SET-POINT-COMMENT

```
(IL:LAMBDA (POINT CONTEXT NODE INDEX OFFSET ITEM TYPE COMPUTE-LOCATION?)
                                                        ; Edited 11-Apr-88 16:46 by woz
```

;;; the SetPoint method for comments

```
    (COND
       ((NULL INDEX)
        (SETQ INDEX (AND OFFSET (IL:NCHARS (CADDR (IL:|fetch| STRUCTURE IL:|of| NODE))))))
       (T (SETQ ITEM (IL:NTH (IL:|fetch| LINEAR-FORM IL:|of| NODE)
                             (IL:ADD1 INDEX)))
          (COND
             ((IL:LISTP ITEM)
              (IF (IL:|type?| WEAK-LINK (CAR ITEM))
                  (SETQ ITEM (IL:|fetch| DESTINATION IL:|of| (CAR ITEM)))
                  (SETQ ITEM (IL:|fetch| DESTINATION IL:|of| (CADR ITEM))))
              (SETQ INDEX (COMMENT-LENGTH NODE (IL:SUB1 (IL:|fetch| SUB-NODE-INDEX IL:|of| ITEM)))))
             (T (SETQ INDEX 0)))))
    (COND
       (INDEX (IL:|replace| POINT-NODE IL:|of| POINT IL:|with| NODE)
              (IL:|replace| POINT-INDEX IL:|of| POINT IL:|with| INDEX)
              (IL:|replace| POINT-TYPE IL:|of| POINT IL:|with| 'STRING)
              (WHEN COMPUTE-LOCATION? (COMPUTE-POINT-POSITION-COMMENT POINT CONTEXT)))
       (T (SET-POINT-NOWHERE POINT)))))
```

(SET-POINT-COMMENT-WORD

```
(IL:LAMBDA (POINT CONTEXT NODE INDEX OFFSET ITEM TYPE COMPUTE-LOCATION?)
                                                        ; Edited 7-Jul-87 11:12 by DCB
    (IL:REPLACE POINT-NODE IL:OF POINT IL:WITH (IL:FETCH SUPER-NODE IL:OF NODE))
    (IL:REPLACE POINT-INDEX IL:OF POINT IL:WITH (IL:IPLUS (COMMENT-LENGTH (IL:FETCH SUPER-NODE IL:OF NODE)
                                                                          (IL:SUB1 (IL:FETCH SUB-NODE-INDEX IL:OF NODE)))
                                                          (SIMPLE-STRING-SCAN (CAR (IL:FETCH LINEAR-FORM
                                                                                             IL:OF NODE))
                                                                              OFFSET T)))
    (IL:REPLACE POINT-TYPE IL:OF POINT IL:WITH 'STRING)
    (WHEN COMPUTE-LOCATION? (COMPUTE-POINT-POSITION-COMMENT POINT CONTEXT))))
```

(SET-SELECTION-COMMENT

```
(IL:LAMBDA (SELECTION CONTEXT NODE INDEX OFFSET ITEM TYPE)     ; Edited 17-Nov-87 11:54 by DCB
```

;;; the SetSelection method for comments

```
    (IF (IL:TYPE? STRING-ITEM ITEM)
        (SET-SELECTION-ME SELECTION CONTEXT NODE)
        (SET-SELECTION-NOWHERE SELECTION))))
```

(SET-SELECTION-COMMENT-WORD

```
(IL:LAMBDA (SELECTION CONTEXT NODE INDEX OFFSET ITEM TYPE)     ; Edited 7-Jul-87 11:12 by DCB
    (IL:REPLACE SELECT-NODE IL:OF SELECTION IL:WITH (IL:FETCH SUPER-NODE IL:OF NODE))
    (IL:REPLACE SELECT-START IL:OF SELECTION IL:WITH (IL:IPLUS (COMMENT-LENGTH (IL:FETCH SUPER-NODE IL:OF NODE)
                                                                               (IL:SUB1 (IL:FETCH SUB-NODE-INDEX
                                                                                                  IL:OF NODE)))
                                                               (SIMPLE-STRING-SCAN (CAR (IL:FETCH LINEAR-FORM
                                                                                                  IL:OF NODE))
                                                                                   OFFSET)))
    (IL:REPLACE SELECT-END IL:OF SELECTION IL:WITH NIL)
    (IL:REPLACE SELECT-TYPE IL:OF SELECTION IL:WITH 'STRING)
    (COMPUTE-SELECTION-POSITION-COMMENT SELECTION CONTEXT)))
```

(SIMPLE-STRING-OFFSET

```
(IL:LAMBDA (STRINGITEM INDEX)                                    ; Edited 17-Nov-87 11:54 by DCB

;;; compute the width of the first index characters in this stringitem.  PRIN2?  is assumed to be false!
    ;; (bind (font _ (fetch Font of stringitem)) (string _ (fetch String of stringitem)) for i from 1 to index sum (CHARWIDTH (NTHCHARCODE string i)
    ;; font))
    (IF (IL:IGREATERP INDEX 0)
        (STRINGWIDTH (IL:SUBSTRING (IL:FETCH STRING IL:OF STRINGITEM)
                                   1 INDEX)
                     (IL:FETCH FONT IL:OF STRINGITEM))
        0)))
```

(SIMPLE-STRING-SCAN

```
(IL:LAMBDA (STRINGITEM OFFSET POINT?)                            ; Edited  7-Jul-87 11:13 by DCB
  (IL:BIND (STRING IL:_ (IL:FETCH STRING IL:OF STRINGITEM))
           (FONT IL:_ (IL:FETCH FONT IL:OF STRINGITEM))
           (INDEX IL:_ 0)
           LENGTH CWIDTH IL:FIRST (IL:SETQ LENGTH (IL:NCHARS STRING)) IL:WHILE (IL:ILEQ (IL:SETQ INDEX
                                                                                              (IL:ADD1 INDEX))
                                                                                    LENGTH)
     IL:DO (IL:SETQ CWIDTH (IL:NTHCHARCODE STRING INDEX))
           (IL:SETQ CWIDTH (IF (IL:ILESSP CWIDTH 32)
                               (IL:IPLUS (IL:CHARWIDTH (IL:CHARCODE IL:^)
                                                       FONT)
                                         (IL:CHARWIDTH (IL:IPLUS 64 CWIDTH)
                                                       FONT))
                               (IL:CHARWIDTH CWIDTH FONT)))
           (IF POINT?
               (WHEN (IL:ILEQ OFFSET (IL:HALF CWIDTH))
                   (RETURN (IL:SUB1 INDEX)))
               (WHEN (IL:ILEQ OFFSET CWIDTH)
                     (RETURN INDEX)))
           (IL:SETQ OFFSET (IL:IDIFFERENCE OFFSET CWIDTH))
     IL:FINALLY (RETURN (IL:SUB1 INDEX)))))
```

(START-COMMENT

```
(IL:LAMBDA (CONTEXT CHARCODE)                                    ; Edited 24-Nov-87 10:22 by DCB
  (LET* ((CARET-POINT (IL:FETCH CARET-POINT IL:OF CONTEXT))
         (POINT-NODE (IL:FETCH POINT-NODE IL:OF CARET-POINT))
         (POINT-TYPE (IL:FETCH POINT-TYPE IL:OF CARET-POINT))
         NEW-NODE)
        (COND
           ((NULL POINT-NODE)
            NIL)
           ((EQ POINT-TYPE 'STRING)
            (WHEN (AND (IL:TYPE? EDIT-NODE POINT-NODE)
                       (EQ (IL:FETCH NODE-TYPE IL:OF POINT-NODE)
                           TYPE-COMMENT)
                       (EQ 0 (IL:FETCH POINT-INDEX IL:OF CARET-POINT)))
                 (UPGRADE-COMMENT CONTEXT POINT-NODE)
                 T))
           ((EQ POINT-TYPE 'ESC-ATOM)
            NIL)
           (T (WHEN (IL:TYPE? EDIT-SELECTION POINT-NODE)
                  (IL:SETQ POINT-NODE (IF (IL:FETCH SELECT-START IL:OF POINT-NODE)
                                          (IL:FETCH SELECT-NODE IL:OF POINT-NODE)
                                          (IL:FETCH SUPER-NODE IL:OF (IL:FETCH SELECT-NODE IL:OF POINT-NODE)))))
              (COND
                 ((EQ (IL:FETCH NODE-TYPE IL:OF POINT-NODE)
                      TYPE-QUOTE)
                  T)
                 (T (INSERT CARET-POINT CONTEXT (LIST (IL:SETQ NEW-NODE (CREATE-NEW-COMMENT CONTEXT))))
                    (WHEN (NOT (DEAD-NODE? NEW-NODE))
                        (IL:REPLACE POINT-NODE IL:OF CARET-POINT IL:WITH NEW-NODE)
                        (IL:REPLACE POINT-INDEX IL:OF CARET-POINT IL:WITH 0)
                        (IL:REPLACE POINT-TYPE IL:OF CARET-POINT IL:WITH 'STRING)
                        (SET-SELECTION-NOWHERE (IL:FETCH SELECTION IL:OF CONTEXT)))
                    T)))))))
```

(STRINGIFY-COMMENT

```
(IL:LAMBDA (NODE ENVIRONMENT)                                    ; Edited 23-Feb-88 11:18 by raf
  (LET ((LEVEL (IL:|fetch| UNASSIGNED IL:|of| NODE)))
       (ECASE LEVEL
           ((0 1 2 3 4) (IL:CONCAT (CADR (IL:|fetch| STRUCTURE IL:|of| NODE))
                                   " "
                                   (CADDR (IL:|fetch| STRUCTURE IL:|of| NODE))))
           (5 (IL:CONCAT (IL:|fetch| (STRING-ITEM STRING) IL:|of| (IL:LISTGET (IL:|fetch| COMMENT-STRING IL:|of|
                                                                                                  ENVIRONMENT
                                                                                                  )
                                                                              5))
                         (CADDR (IL:|fetch| STRUCTURE IL:|of| NODE))
                         (IL:|fetch| (STRING-ITEM STRING) IL:|of| (IL:LISTGET (IL:|fetch| COMMENT-STRING IL:|of| ENVIRONMENT)
```

```
                                                                    6)))))))
```

(CREATE-COMMENT-WORD-NODE

```
  (IL:LAMBDA (CHARS ENVIRONMENT)                                    ; Edited 13-Apr-88 14:51 by woz
    (CREATE-SIMPLE-NODE CHARS ENVIRONMENT TYPE-COMMENT-WORD CHARS NIL (IL:|fetch| COMMENT-FONT IL:|of| ENVIRONMENT))))
```

(CREATE-COMMENT-WORD-NODES

```
  (IL:LAMBDA (CHARS SUBNODES ENVIRONMENT)                           ; Edited  7-Jul-87 11:13 by DCB
    (IL:|bind| (END IL:_ (IL:NCHARS CHARS))
          I OK? IL:|first| (IL:SETQ I END) IL:|while| (IL:NEQ I 0)
       IL:|do| (COND
                  ((IL:NEQ (IL:NTHCHARCODE CHARS I)
                           (IL:CHARCODE IL:SP))
                   (IL:SETQ OK? T))
                  (OK? (PUSH (CREATE-COMMENT-WORD-NODE (IL:SUBSTRING CHARS (IL:ADD1 I)
                                                                     END)
                                     ENVIRONMENT)
                             SUBNODES)
                       (IL:SETQ END I)
                       (IL:SETQ OK? NIL)))
               (IL:SETQ I (IL:SUB1 I)))
       IL:|finally| (RETURN (CONS (CREATE-COMMENT-WORD-NODE (IL:SUBSTRING CHARS 1 END)
                                        ENVIRONMENT)
                                  SUBNODES)))))
```

(UNDO-COMMENT-CHANGE

```
  (IL:LAMBDA (CONTEXT NODE OLD-VALUE)                               ; Edited 13-Apr-88 15:31 by woz
    (UNDO-BY UNDO-COMMENT-CHANGE NODE (CADDR (IL:FETCH STRUCTURE IL:OF NODE)))
    (LET ((COMMENT-WORDS (PARSE-STRING-INTO-WORDS OLD-VALUE))
          (SUBNODES (IL:|fetch| SUB-NODES IL:|of| NODE)))
         (RPLACA (CDDR (IL:|fetch| STRUCTURE IL:|of| NODE))
                 OLD-VALUE)
         (IL:|for| WORD IL:|in| COMMENT-WORDS IL:|as| SUB-NODE-INDEX IL:|from| 1
            IL:|do| (COND
                       ((CDR SUBNODES)
                        (IL:|replace| STRUCTURE IL:|of| (CADR SUBNODES) IL:|with| WORD)
                        (NOTE-CHANGE-IN-SIMPLE (CADR SUBNODES)
                               CONTEXT))
                       (T (IL:NCONC1 SUBNODES (CREATE-SIMPLE-NODE WORD (IL:|fetch| ENVIRONMENT IL:|of| CONTEXT)
                                                                  TYPE-COMMENT-WORD WORD NIL (IL:|fetch| COMMENT-FONT
                                                                                                  IL:|of| (IL:|fetch| ENVIRONMENT
                                                                                                                      IL:|of| CONTEXT))))
                          (IL:|replace| SUPER-NODE IL:|of| (CADR SUBNODES) IL:|with| NODE)
                          (IL:|replace| SUB-NODE-INDEX IL:|of| (CADR SUBNODES) IL:|with| SUB-NODE-INDEX)))
                    (IL:SETQ SUBNODES (CDR SUBNODES))
            IL:|finally|                                            ; throw away extra subnodes
                    (RPLACD SUBNODES)
                    (RPLACA (IL:|fetch| SUB-NODES IL:|of| NODE)
                            (IL:FLENGTH COMMENT-WORDS)))
         (NOTE-CHANGE NODE CONTEXT))))
```

(UPGRADE-COMMENT

```
  (IL:LAMBDA (CONTEXT NODE)                                         ; Edited  7-Jul-87 11:13 by DCB
    (WHEN (IL:ILESSP (IL:FETCH UNASSIGNED IL:OF NODE)
                     (IL:CONSTANT (IL:LENGTH COMMENT-MARKERS)))
          (RPLACA (CDR (IL:FETCH STRUCTURE IL:OF NODE))
                  (CAR (IL:NTH COMMENT-MARKERS (IL:ADD (IL:FETCH UNASSIGNED IL:OF NODE)
                                                       1))))
          (NOTE-CHANGE NODE CONTEXT)
          (WHEN (IL:FETCH SUPER-NODE IL:OF (IL:FETCH SUPER-NODE IL:OF NODE))
                                                                    ; this node has a supernode that is not the root
              (NOTE-CHANGE (IL:FETCH SUPER-NODE IL:OF NODE)
                           CONTEXT))
          (UNDO-BY DEGRADE-COMMENT NODE))))

)
```

(DEFUN MAKE-COMMENT-STRING (NODE)

;;; get the comment words from the subnodes and put them together into one string (as efficiently as possible)

```
   (LET* ((SUBNODES (CDR (IL:|fetch| SUB-NODES IL:|of| NODE)))
          (LENGTH (LET ((SUM 0))
                       (DOLIST (SUBNODE SUBNODES SUM)
                           (INCF SUM (LENGTH (IL:|fetch| STRUCTURE IL:|of| SUBNODE))))))
          (STRING (MAKE-STRING LENGTH))
          (POINTER 0))
         (DOLIST (SUBNODE SUBNODES STRING)
             (LET ((WORD (IL:|fetch| STRUCTURE IL:|of| SUBNODE)))
                  (REPLACE STRING WORD :START1 POINTER)
                  (INCF POINTER (LENGTH WORD))))))
```

(DEFUN **VERIFY-COMMENT** (NODE)

;;; check the comment in this node the strings in the subnodes (ie verify-comment). return T if they match, NIL otherwise.

```
   (LET* ((POINTER 0)
          (STRING (THIRD (IL:FETCH STRUCTURE IL:OF NODE)))
          (STRING-LENGTH (LENGTH STRING)))
         (DOLIST (SUBNODE (CDR (IL:|fetch| SUB-NODES IL:|of| NODE))
                          T)
             (LET* ((WORD (IL:|fetch| STRUCTURE IL:|of| SUBNODE))
                    (WORD-LENGTH (LENGTH WORD)))
                   (WHEN (MISMATCH STRING WORD :START1 POINTER :END1 (MIN (INCF POINTER (LENGTH WORD))
                                                                          STRING-LENGTH))
                         (RETURN NIL))))))
```

(IL:PUTPROPS **IL:SEDIT-COMMENTS IL:COPYRIGHT** ("Venue & Xerox Corporation" 1986 1987 1988 1990))

FUNCTION INDEX

CONSTANT INDEX

PROPERTY INDEX