File created: 5-Nov-2020 19:59:11 **{DSK}<users>arunwelch>skydrive>documents>unix>lisp>lde>notecards>system>NCCOMPACT.;5**

previous date: 9-Jan-94 18:56:38 {DSK}<users>arunwelch>skydrive>documents>unix>lisp>lde>notecards>system>NCCOMPACT.;4

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ NCCOMPACTCOMS
       (                                                       ; File created by KELLEY


;;; NoteFile compactor.

        (FNS NC.GetCardPartLength NC.SortIndexEntries NC.UpdateCardLoc)


;;; Compact to target

        (FNS NC.CompactNoteFileToTarget NC.CopySortedCardPart NC.CopySortedCardParts
             NC.ComputeNewDatabaseIndexSize)


;;; Compact in place

        (FNS NC.CompactNoteFileInPlace NC.CopySortedCardPartInPlace)
        (PROP (FILETYPE MAKEFILE-ENVIRONMENT)
              NCCOMPACT)))

;; File created by KELLEY

;;; NoteFile compactor.

(DEFINEQ

(NC.GetCardPartLength
  [LAMBDA (Card FromPtr)                                        (* kirk%: " 2-Jan-86 06:36")
                                                                (* Comment)
    (LET ((NoteFile (fetch (Card NoteFile) of Card)))
         (SETFILEPTR (fetch (NoteFile Stream) of NoteFile)
                FromPtr)
         (FIXP (NC.ReadPtr (fetch (NoteFile Stream) of NoteFile)
                      3])

(NC.SortIndexEntries
  [LAMBDA (NoteFile)                                            (* kirk%: " 6-Aug-86 14:13")

          (* * Using the IndexArray for Stream, return the sorted list of index entries as triples of fileptr, ID, and EntryType.)

          (* * fkr 11/8/85%: Now uses notefile object and hash arrays instead of index arrays.)

          (* * kirk 8Dec85 Took out PROGN around FUNCTION returning CollectionList)

          (* * If we add any card parts must change inference of total active cards in NC.CompactNoteFile)

    (LET (CardParts)
         [NC.MapCards NoteFile (FUNCTION (LAMBDA (Card)
                                           (if (EQ (fetch (Card Status) of Card)
                                                   'ACTIVE)
                                               then (push CardParts (create SortingRecord
                                                                           FileLoc _ (fetch (Card MainLoc)
                                                                                        of Card)
                                                                           Card _ Card
                                                                           CardPartTypeNum _ 0))
                                                    (push CardParts (create SortingRecord
                                                                           FileLoc _ (fetch (Card LinksLoc)
                                                                                        of Card)
                                                                           Card _ Card
                                                                           CardPartTypeNum _ 1))
                                                    (push CardParts (create SortingRecord
                                                                           FileLoc _ (fetch (Card TitleLoc)
                                                                                        of Card)
                                                                           Card _ Card
                                                                           CardPartTypeNum _ 2))
                                                    (push CardParts (create SortingRecord
                                                                           FileLoc _ (fetch (Card PropListLoc)
                                                                                        of Card)
                                                                           Card _ Card
                                                                           CardPartTypeNum _ 3]
         (SORT CardParts T])

(NC.UpdateCardLoc
  [LAMBDA (Card CardPartTypeNum ToPtr)                             (* rht%: " 2-May-87 16:34")

          (* * rht 5/2/87%: Now returns NIL if bad notefile.)

    (SELECTQ CardPartTypeNum
         (0 (replace (Card MainLoc) of Card with ToPtr))
         (1 (replace (Card LinksLoc) of Card with ToPtr))
         (2 (replace (Card TitleLoc) of Card with ToPtr))
         (3 (replace (Card PropListLoc) of Card with ToPtr))
         (PROGN (FLASHW PROMPTWINDOW)
                (NC.PrintMsg NIL T "Bad NoteFile. Please Inspect and Repair." (CHARACTER 13))
               NIL])
)


;;; Compact to target

(DEFINEQ

(NC.CompactNoteFileToTarget
  [LAMBDA (FromNoteFile ToFileName InterestedWindow)               ; Edited  3-Dec-87 18:59 by rht:

          (* * In sorted order, copy card parts to lower addresses in the file.)

          (* * fgh |5/1/86| Now returns the ToNoteFile in order to be compatible with compact in place.)

          (* * rht 11/3/86%: Now opens FromNoteFile read-only. Also now takes InterestedWindow arg.)

          (* * rht 1/22/87%: Slight change to computation of new index size.)

          (* * rht 3/17/87%: Added RESETLST to make sure notefiles get closed in case of bombing out.)

          (* * rht 5/15/87%: No longer calls NC.ComputeNewDatabaseIndexSize.
          Target notefile's index will be same size as source notefile's.)

          (* * pmi 5/27/87%: Removed HashArray argument in calls to NC.OpenNoteFile.
          Added call to NC.NoticeNoteFile to notice the original and target notefiles.
          Also stopped creation of a notefile interface for the target notefile before compaction -
          it should be done at the end of compaction instead.)

          (* * pmi 6/24/87%: Now returns NIL if can't create the target notefile.)

          (* * pmi 6/25/87%: Now passes NIL for Can'tTruncateFlg in call to NC.OpenNoteFile.)

          (* * pmi 8/13/87%: Removed calls to NC.NoticeNoteFile; they are now done further up in NC.CompactNoteFile.)

    (PROG (ToNoteFile OperationMsg)
          (if (SETQ FromNoteFile (NC.OpenNoteFile FromNoteFile T T T NIL NIL T T InterestedWindow NIL NIL NIL T))
              then (SETQ OperationMsg (CONCAT "Compacting " (fetch (NoteFile FullFileName) of FromNoteFile)
                                             (CHARACTER 13)))
                   (SETQ ToNoteFile (NC.CreateDatabaseFile ToFileName (fetch (NoteFile HashArraySize)
                                                                         of FromNoteFile)
                                           OperationMsg T NIL T InterestedWindow NIL NIL NIL NIL T))
                   (if (EQ ToNoteFile 'CreateCancelled)
                       then (RETURN NIL)
                     else (SETQ ToNoteFile
                                (NC.OpenNoteFile ToNoteFile T T T T T T T InterestedWindow NIL NIL NIL NIL NIL T)))
                   (RESETLST
                       [RESETSAVE NIL `(NC.ForceDatabaseClose ,FromNoteFile]
                       [RESETSAVE NIL `(NC.ForceDatabaseClose ,ToNoteFile]
                       (LET ((OriginalStream (fetch (NoteFile Stream) of FromNoteFile))
                             (TargetStream (fetch (NoteFile Stream) of ToNoteFile))
                             FromFileLength TargetFileLength BytesRecovered)
                            (replace (NoteFile NextIndexNum) of ToNoteFile with (fetch (NoteFile NextIndexNum)
                                                                                     of FromNoteFile))
                            (SETFILEPTR TargetStream (NC.TotalIndexSize (fetch (NoteFile HashArraySize)
                                                                               of ToNoteFile)))
                                                                 (* truncate ToNoteFile after the index)
                            (if (NC.CopySortedCardParts (NC.SortIndexEntries FromNoteFile)
                                      ToNoteFile NIL NIL NIL InterestedWindow OperationMsg)
                                then                             (* all useable card parts got copied)
                                     (SETQ FromFileLength (GETEOFPTR OriginalStream))

          (* * fool NC.PutHashArray into writing out the index for the new NoteFile)

                                     (replace (NoteFile Stream) of FromNoteFile with TargetStream)
                                     (NCLocalDevice.PutHashArray FromNoteFile InterestedWindow T OperationMsg)
                                     (replace (NoteFile Stream) of FromNoteFile with OriginalStream)

          (* * Put out the new ChkptPtr to the file.)
```

```
                                     (replace (NoteFile CheckptPtr) of ToNoteFile with (SETQ TargetFileLength
                                                                                       (GETEOFPTR TargetStream)))

          (* * Steal the UID from the original file so links will work. Write out the header.)

                                     (replace (NoteFile UID) of ToNoteFile with (fetch (NoteFile UID) of FromNoteFile
                                                                                     ))
                                     (NC.PutNoteFileHeader ToNoteFile)
                                     (SETQ BytesRecovered (DIFFERENCE FromFileLength TargetFileLength))
                                     (NC.PrintMsg NIL T (fetch (NoteFile FullFileName) of FromNoteFile)
                                            " compacted to "
                                            (fetch (NoteFile FullFileName) of ToNoteFile)
                                            (CHARACTER 13)
                                            "Recovered " BytesRecovered " bytes (" (FIX (TIMES 100
                                                                                    (FQUOTIENT
                                                                                        BytesRecovered

                                                                                        FromFileLength
                                                                                          )))
                                            "%%)"
                                            (CHARACTER 13))
                                     (NC.ClearMsg InterestedWindow T))))
                   (RETURN ToNoteFile])

(NC.CopySortedCardPart
  [LAMBDA (SortingRecord ToNoteFile InterestedWindow)              (* rht%: " 1-May-87 14:46")

          (* * Copy some portion of a card; title, links, substance, proplist, to ptr in Stream and update index accordingly.)

          (* * fkr 11/8/85%: No longer worries about link labels part. Also updated to hadle new CradID and NoteFileobject.)

          (* * kirk 23Nov85%: restructured to reduce redundant code)

          (* * rht 1/22/87%: Added InterestedWindow arg.)

          (* * rht 5/1/87%: Changed mention of FromNoteFile to ToNoteFile.)

    (WITH.MONITOR (NC.FetchMonitor ToNoteFile)
        (LET ((FromPtr (fetch (SortingRecord FileLoc) of SortingRecord))
              (Card (fetch (SortingRecord Card) of SortingRecord))
              (CardPartTypeNum (fetch (SortingRecord CardPartTypeNum) of SortingRecord))
              (ToStream (fetch (NoteFile Stream) of ToNoteFile))
              ToPtr)
             (SETQ ToPtr (GETFILEPTR ToStream))
             (if (NC.CopyCardPart (fetch (NoteFile Stream) of (fetch (Card NoteFile) of Card))
                       ToStream FromPtr ToPtr InterestedWindow)
                 then (NC.UpdateCardLoc Card CardPartTypeNum ToPtr))))])

(NC.CopySortedCardParts
  [LAMBDA (SortedSortingRecords NewNoteFile ToPtr BufferStream OriginalLengthOfFile InterestedWindow
                 OperationMsg)                                       (* rht%: "22-Jan-87 23:22")

          (* * Copies card parts to a note file. Assumes FILEPTR is set to the right location in the destination file.
          If NewNoteFile is NIL, then ToPtr BuferStream and OriginalLengthOfFile are used to CopySortedCardPartInPlace.)

          (* * rht 11/3/86%: Added InterestedWindow and OperationMsg arg.)

          (* * rht 1/22/87%: Now passes InterestedWindow to NC.CopySortedCardPartInPlace and NC.CopySortedCardPart.)

    [OR OperationMsg (SETQ OperationMsg (CONCAT "Compacting NoteFile" (CHARACTER 13]
    (LET ((TotalUseableCardParts (LENGTH SortedSortingRecords)))
         (NC.PrintMsg InterestedWindow T OperationMsg "Copying piece " 1 " of " TotalUseableCardParts "."
                (CHARACTER 13))
         (for SortedRecord in SortedSortingRecords as CTR from 1 eachtime (BLOCK)
            do (if (ZEROP (IREMAINDER CTR 20))
                   then (NC.PrintMsg InterestedWindow T OperationMsg "Copying piece " CTR " of "
                              TotalUseableCardParts "." (CHARACTER 13)))
               (if [NOT (SETQ ToPtr (if NewNoteFile
                                        then (NC.CopySortedCardPart SortedRecord NewNoteFile InterestedWindow)
                                      else (NC.CopySortedCardPartInPlace SortedRecord BufferStream ToPtr
                                                 OriginalLengthOfFile InterestedWindow]
                   then (RETURN NIL))
            finally (RETURN ToPtr])

(NC.ComputeNewDatabaseIndexSize
  [LAMBDA (NoteFile)                                                 (* rht%: "22-Jan-87 21:45")

          (* If the number of notecards we have is more than |3/4| the size of the index, double the index size;
          otherwise just keep the same size)

          (* * rht 1/22/87%: Now returns NIL if index doesn't need expanding.)

    (LET ((HashArraySize (fetch (NoteFile HashArraySize) of NoteFile)))
         (if (GREATERP (fetch (NoteFile NextIndexNum) of NoteFile)
                     (FIX (TIMES 0.75 HashArraySize)))
              then (TIMES 2 HashArraySize])

)


;;; Compact in place

(DEFINEQ

(NC.CompactNoteFileInPlace
  [LAMBDA (NoteFile InterestedWindow)                                 ; Edited 31-Dec-87 21:43 by Trigg

    ;; Compact the notefile in place.  If index needs to be increased, then first make room for bigger index by copying.  Compaction is done by sorting
    ;; index pointers and moving each entry in the file to lower addresses.

    ;; fkr 11/8/85 Updated to handle new CardID scheme and NoteFile object.

    ;; kirk 19Nov85: Created from NC.CompactDatabaseInPlace

    ;; rht 11/3/86: Added InterestedWindow arg.

    ;; rht&pmi 12/9/86: Moved RESETLST to be after call to NC.OpenDatabaseFile.

    ;; rht 1/22/87: Now bails out if NC.ExpandIndexInPlace was called but returned NIL.  Also fixed message when size of notefile actually increases.

    ;; rht 5/16/87: No longer calls NC.ExpandIndexInPlace.

    ;; pmi 5/20/87: Removed HashArray argument in calls to NC.OpenNoteFile.

    ;; rht 12/31/87: Changed call to SETFILEINFO to pass Stream argument rather than file name.

    (DECLARE (GLOBALVARS PROMPTWINDOW))
    (if (AND NoteFile (SETQ NoteFile (NC.OpenNoteFile NoteFile T T NIL NIL T T T InterestedWindow)))
        then (RESETLST
                  [RESETSAVE NIL `(NC.ForceDatabaseClose ,NoteFile]
                  (LET ((SourceStream (fetch (NoteFile Stream) of NoteFile))
                        [BufferStream (OPENSTREAM '{NODIRCORE} 'BOTH NIL '((TYPE BINARY]
                        (FullFileName (fetch (NoteFile FullFileName) of NoteFile))
                        ToPtr OriginalLengthOfFile OperationMsg)

                       ;; Expand index if needed.

                       (SETQ OperationMsg (CONCAT "Compacting " FullFileName " in place." (CHARACTER 13)))
                       (SETQ OriginalLengthOfFile (GETEOFPTR SourceStream))

                       ;; In sorted order, copy entries to lower locations in the file.  Expand index first if necessary.

                       (if (SETQ ToPtr (NC.CopySortedCardParts (NC.SortIndexEntries NoteFile)
                                               NIL
                                               (NC.TotalIndexSize (fetch (NoteFile HashArraySize) of NoteFile))
                                               BufferStream
                                               (GETEOFPTR SourceStream)
                                               InterestedWindow OperationMsg))
                           then                                       ; all useable card parts got copied
                                (NC.PutCheckptPtr NoteFile ToPtr)     ; Put out the new ChkptPtr to the file.

                                ;; Truncate file at that point.

                                (NC.PrintMsg InterestedWindow T "Truncating file " FullFileName " ...")
                                (NCLocalDevice.PutHashArray NoteFile InterestedWindow T OperationMsg)
                                (if (NOT (SETFILEINFO SourceStream 'LENGTH ToPtr))
                                    then (NC.PrintMsg InterestedWindow NIL "Couldn't truncate " FullFileName "."
                                                (CHARACTER 13))
                                  else (NC.PrintMsg InterestedWindow T "Done." (CHARACTER 13))
                                       (NC.ClearMsg InterestedWindow T))
                                [LET ((NumBytesSaved (DIFFERENCE OriginalLengthOfFile ToPtr)))
                                      (if (MINUSP NumBytesSaved)
                                          then (NC.PrintMsg NIL T FullFileName " compacted in place." (CHARACTER
                                                                                                         13)
                                                       "Increased by "
                                                       (MINUS NumBytesSaved)
                                                       " bytes ("
                                                       (FIX (TIMES 100 (FQUOTIENT (MINUS NumBytesSaved)
                                                                               OriginalLengthOfFile)))
                                                       "%%)"
                                                       (CHARACTER 13))
                                        else (NC.PrintMsg NIL T FullFileName " compacted in place." (CHARACTER 13)
                                                    "Recovered " NumBytesSaved " bytes ("
                                                    (FIX (TIMES 100 (FQUOTIENT NumBytesSaved OriginalLengthOfFile)))
                                                    "%%)"
                                                    (CHARACTER 13]
                                (SETQ NC.DatabaseFileNameSuggestion (PACKFILENAME 'VERSION NIL 'BODY FullFileName)
                                 )
                         else (FLASHW PROMPTWINDOW)
                              (NC.PrintMsg NIL T "Compact of " FullFileName " cancelled."))))])

(NC.CopySortedCardPartInPlace
  [LAMBDA (SortingRecord BufferStream ToPtr EOFBufferLoc InterestedWindow)
                                                                      (* rht%: " 2-May-87 16:36")

           (* * Copy some portion of a card; title, links, substance, proplist, to ptr in Stream and update index accordingly.)

           (* * fkr 11/8/85%: No longer worries about link labels part. Also updated to hadle new CradID and NoteFileobject.)
```

```
          (* * kirk 23Nov85%: added ToNoteFile parameter and restructured to reduce redundant code and to blindly COPYBYTES)

          (* * rht 1/22/87%: Now properly handles case when NC.CopyCardPart returns NIL.
          This function no longer called under NC.ExpandIndexInPlace. Added InterestedWindow arg.)

          (* * rht 5/2/87%: Now returns nil if last call to NC.UpdateCardLoc fails.)

    (LET ((FromPtr (fetch (SortingRecord FileLoc) of SortingRecord))
          (Card (fetch (SortingRecord Card) of SortingRecord))
          (CardPartTypeNum (fetch (SortingRecord CardPartTypeNum) of SortingRecord))
          NoteFile)
         (if (EQUAL ToPtr FromPtr)
             then (PLUS ToPtr (NC.GetCardPartLength Card FromPtr))
           else (WITH.MONITOR (NC.FetchMonitor (SETQ NoteFile (fetch (Card NoteFile) of Card)))
                    (LET ((Stream (fetch (NoteFile Stream) of NoteFile))
                          EndPtr Length)

          (* * Copy the substance out to the {NODIRCORE} stream.)

                         (SETQ Length (NC.CopyCardPart Stream BufferStream FromPtr 0 InterestedWindow))

          (* * Copy to end of file if needed for safety. That way, if we crash during second copy, all nearby card parts are still okay.)

                         (SETQ EndPtr (PLUS ToPtr Length))
                         (if (OR (GEQ FromPtr EndPtr)
                                 (AND (NC.CopyCardPart BufferStream Stream 0 EOFBufferLoc InterestedWindow)
                                      (NC.UpdateCardLoc Card CardPartTypeNum EOFBufferLoc)))
                             then

          (* * Now copy to its proper home.)

                                  (AND (NC.CopyCardPart BufferStream Stream 0 ToPtr InterestedWindow)
                                       (NC.UpdateCardLoc Card CardPartTypeNum ToPtr)
                                       EndPtr))))])
)

(PUTPROPS NCCOMPACT FILETYPE :FAKE-COMPILE-FILE)

(PUTPROPS NCCOMPACT MAKEFILE-ENVIRONMENT (:PACKAGE "IL" :READTABLE "INTERLISP" :BASE 10))

(PUTPROPS NCCOMPACT COPYRIGHT ("Venue & Xerox Corporation" 1985 1986 1987 1988 1989 1990 1993 1994 2020))
```

FUNCTION INDEX

PROPERTY INDEX