File created: 15-Jun-90 13:42:42 **{DSK}<usr>local>lde>lispcore>internal>library>datepatch.;2**

changes to: (VARS DATEPATCHCOMS)

previous date: 30-May-89 12:29:12 {DSK}<usr>local>lde>lispcore>internal>library>datepatch.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS


```
(RPAQQ DATEPATCHCOMS
       (;; Patches to the date parser and printer:  IDATE parses many more dates now (full month names, more time zones).  GDATE handles
        ;; timezones outside of US.
        (FNS IDATE \IDATESCANTOKEN \IDATE-PARSE-MONTH \OUTDATE \OUTDATE-STRING \UNPACKDATE)
        (VARS TIME.ZONES)
        (DECLARE%: EVAL@COMPILE DONTCOPY (CONSTANTS \4YearsDays)
               (LOCALVARS . T)
               (GLOBALVARS TIME.ZONES \TimeZoneComp \DayLightSavings)
               (SPECVARS *STR* *POS*))))
```

;; Patches to the date parser and printer: IDATE parses many more dates now (full month names, more time zones). GDATE handles timezones outside of US.

```
(DEFINEQ

(IDATE
  [LAMBDA (STR DEFAULTTIME)                                  ; Edited  4-May-89 18:22 by bvm
    (if (NULL STR)
        then (DAYTIME)
      else
      (PROG ((*STR* (MKSTRING STR))
             (*POS* 1)
             MONTH DAY YEAR HOUR MINUTES SECONDS N1 N2 CH DLS TIMEZONE)
            (DECLARE (CL:SPECIAL *STR* *POS*))
        TOP (OR (SETQ N1 (\IDATESCANTOKEN))
                (RETURN NIL))
            (SELCHARQ (NTHCHARCODE *STR* *POS*)
                 ((/ - SPACE)                                ; Okay to put inside date
                       (add *POS* 1))
                 ("," (if (LISTP N1)
                          then                               ; Assume str was something like Mon, Apr 1....  Trash the day.
                               (add *POS* 1)
                               (GO TOP)))
                 ("." (if (LISTP N1)
                          then                               ; Abbreviated month?
                               (add *POS* 1)))
                 NIL)
            (OR (SETQ N2 (\IDATESCANTOKEN))
                (RETURN NIL))
            (SELCHARQ (NTHCHARCODE *STR* *POS*)
                 ((/ - SPACE %,)
                       (add *POS* 1))
                 ("." (if (LISTP N2)
                          then                               ; Abbreviated month?
                               (add *POS* 1)))
                 NIL)
            (if [NOT (FIXP (SETQ YEAR (\IDATESCANTOKEN]
                then (RETURN NIL)
              elseif (< YEAR 100)
                then                                         ; default to this century
                     (add YEAR 1900)
              elseif (OR (< YEAR 1900)
                         (> YEAR 2037))
                then                                         ; out of range
                     (RETURN NIL))                           ; Now figure out day and month
            (if (FIXP N2)
                then                                         ; Must be month-day
                     (SETQ DAY N2)
                     (SETQ MONTH N1)
              elseif (FIXP (SETQ DAY N1))
                then                                         ; day-month
                     (SETQ MONTH N2)
              else (RETURN NIL))
            (if (FIXP MONTH)
                then (if (OR (< MONTH 1)
                             (> MONTH 12))
                         then                                ; invalid month
                              (RETURN NIL))
              elseif (SETQ MONTH (\IDATE-PARSE-MONTH MONTH))
              else (RETURN NIL))
```

```
            (if (OR (< DAY 1)
                    (> DAY (SELECTQ MONTH
                                ((9 4 6 11)                          ; 30 days hath September...
                                  30)
                                (2 (if (EVENP YEAR 4)
                                       then 29
                                     else 28))
                                31)))
                then (RETURN NIL))
            (while (EQ (SETQ CH (NTHCHARCODE *STR* *POS*))
                       (CHARCODE SPACE))
               do                                                    ; Skip spaces
                  (add *POS* 1))
            (SELCHARQ (NTHCHARCODE *STR* *POS*)
                 (","                                                ; Ok to terminate date with comma
                      (add *POS* 1))
                 (NIL                                                ; No time.  Ok if DEFAULTTIME passed in
                      (if (NULL DEFAULTTIME)
                          then (RETURN NIL))
                      (SETQ SECONDS (IREMAINDER DEFAULTTIME 60))
                      (SETQ MINUTES (IREMAINDER (SETQ DEFAULTTIME (IQUOTIENT DEFAULTTIME 60))
                                                60))
                      (SETQ HOUR (IQUOTIENT DEFAULTTIME 60))
                      (GO DONE))
                 NIL)
;; Now scan time
            (if [NOT (FIXP (SETQ HOUR (\IDATESCANTOKEN]
                then (RETURN NIL))
            (if (EQ (SETQ CH (NTHCHARCODE *STR* *POS*))
                    (CHARCODE %:))
                then                                                 ; hh:mm
                     (add *POS* 1)
                     (OR (FIXP (SETQ MINUTES (\IDATESCANTOKEN)))
                         (RETURN NIL))
                     (if (EQ (SETQ CH (NTHCHARCODE *STR* *POS*))
                             (CHARCODE %:))
                         then                                        ; hh:mm:ss
                              (add *POS* 1)
                              (OR (FIXP (SETQ SECONDS (\IDATESCANTOKEN)))
                                  (RETURN NIL))
                              (SETQ CH (NTHCHARCODE *STR* *POS*)))
              else                                                   ; break apart time given without colon
                   (SETQ MINUTES (IREMAINDER HOUR 100))
                   (SETQ HOUR (IQUOTIENT HOUR 100)))
            [if CH
                then                                                 ; There's more
                     [while (EQ CH (CHARCODE SPACE)) do              ; Skip spaces
                                                  (SETQ CH (NTHCHARCODE *STR* (add *POS* 1]
                     [if [AND (FMEMB CH (CHARCODE (A P a p)))
                              (FMEMB (NTHCHARCODE *STR* (ADD1 *POS*))
                                     (CHARCODE (M m)))
                              (FMEMB (NTHCHARCODE *STR* (+ *POS* 2))
                                     (CHARCODE (SPACE - NIL]
                         then                                        ; AM or PM appended
                              (if (NOT (< HOUR 13))
                                  then                               ; bogus
                                       (RETURN NIL))
                              (if (EQ HOUR 12)
                                  then                               ; wrap to zero
                                       (SETQ HOUR 0))
                              (if (FMEMB CH (CHARCODE (P p)))
                                  then                               ; PM = 12 hours later
                                       (add HOUR 12))
                              (SETQ CH (NTHCHARCODE *STR* (add *POS* 2)))
                              (while (EQ CH (CHARCODE SPACE)) do     ; Skip spaces
                                                             (SETQ CH (NTHCHARCODE *STR* (add *POS* 1]
                     ;; Now check for time zone
                     [if [AND (EQ CH (CHARCODE -))
                              (ALPHACHARP (NTHCHARCODE *STR* (ADD1 *POS*]
                         then                                        ; Some obsolete date forms gave time zone separated from time
                                                                     ; by hyphen
                              (SETQ CH (NTHCHARCODE *STR* (add *POS* 1]
                     (SELCHARQ CH
                          ((+ -)                                     ; Explicit offset +-hhmm from GMT
                               (add *POS* 1)
                               (if [NOT (FIXP (SETQ TIMEZONE (\IDATESCANTOKEN]
                                   then (RETURN NIL))
                               (CL:MULTIPLE-VALUE-BIND (H M)
                                   (CL:TRUNCATE TIMEZONE 100)
                                 [SETQ TIMEZONE (if (EQ M 0)
                                                    then H
                                                  else               ; Non-hour timezone.  Use ratios.
                                                       (+ H (/ M 60])
                               (if (EQ CH (CHARCODE +))
                                   then                              ; we represent time zones the other way around, so have to
                                                                     ; negate
```

```
                                (SETQ TIMEZONE (- TIMEZONE))))
                        (if (AND CH (ALPHACHARP CH))
                            then                                    ; Perhaps symbolic time zone
                                 (PROG ((START *POS*))
                                   LP  (if [NULL (SETQ CH (NTHCHARCODE *STR* (add *POS* 1]
                                           elseif (ALPHACHARP CH)
                                             then (GO LP)
                                           elseif (EQ CH (CHARCODE SPACE))
                                             then                   ; Space may terminate, except that some time zones have space
                                                                    ; in middle, e.g., EET DST.
                                                  (if (AND (SETQ CH (NTHCHARCODE *STR* (ADD1 *POS*)))
                                                           (ALPHACHARP CH))
                                                      then (add *POS* 1)
                                                           (GO LP))
                                           else                     ; Non-alphabetic in timezone
                                                (RETURN NIL))
                                 ;; Potential time zone from START to before POS
                                      (SETQ TIMEZONE (SUBSTRING *STR* START (SUB1 *POS*)))
                                      (RETURN (SETQ TIMEZONE
                                                (for ZONE in TIME.ZONES bind DST
                                                   do (if (STRING-EQUAL TIMEZONE (CADR ZONE))
                                                          then (RETURN (CAR ZONE))
                                                        elseif (AND (SETQ DST (CADDR ZONE))
                                                                    (STRING-EQUAL TIMEZONE DST))
                                                          then ; The daylight equivalent is off by one hour
                                                               (RETURN (SUB1 (CAR ZONE]
        DONE
             (RETURN (AND (< HOUR 24)
                          (< MINUTES 60)
                          (OR (NOT SECONDS)
                              (< SECONDS 60))
                          (\PACKDATE YEAR (SUB1 MONTH)
                                 DAY HOUR MINUTES (OR SECONDS 0)
                                 TIMEZONE])
```

(\IDATESCANTOKEN

```
[LAMBDA NIL                                                         ; Edited  4-May-89 15:20 by bvm
  (DECLARE (CL:SPECIAL *STR* *POS*))
```

;; Returns next token in STR, starting at POS. Is either an integer or list of alphabetic charcodes. Skips blanks

```
(PROG (RESULT CH)
  LP  (SETQ CH (NTHCHARCODE *STR* *POS*))
      (RETURN (COND
                 ((NULL CH)
                  NIL)
                 ((EQ CH (CHARCODE SPACE))                          ; Skip leading spaces
                  (add *POS* 1)
                  (GO LP))
                 ((DIGITCHARP CH)
                  (SETQ RESULT (- CH (CHARCODE 0)))
                  [while (AND (SETQ CH (NTHCHARCODE *STR* (add *POS* 1)))
                              (DIGITCHARP CH))
                     do (SETQ RESULT (+ (- CH (CHARCODE 0))
                                        (TIMES RESULT 10]
                  RESULT)
                 ((ALPHACHARP CH)
                  (CONS (UCASECODE CH)
                        (while (AND (SETQ CH (NTHCHARCODE *STR* (add *POS* 1)))
                                    (ALPHACHARP CH))
                           collect (UCASECODE CH])
```

(\IDATE-PARSE-MONTH

```
[LAMBDA (MONTH)                                                     ; Edited  4-May-89 14:54 by bvm
```

;; MONTH is a list of upper case character codes. Figure out which month (1-12) we mean. We require that MONTH be at least 3 characters long
;; and a prefix of month name

;; These ugly macros produce code, essentially a decision tree, that walks down the list of char codes looking for exactly the right ones.

```
(CL:MACROLET
 [[DISCRIMINATE (FORMS)
```

;; The entry -- start MINCHARS at 3 and turn the month names into char codes. FORMS is quoted list to workaround masterscope
;; stupidity

```
         `(DISCRIMINATE-1 3 ,@(FOR F IN (CADR FORMS) COLLECT (CONS (CHCON (CAR F))
                                                                   (CDR F]
  [DISCRIMINATE-1 (MINCHARS &BODY FORMS)
         (IF (NULL (CDR FORMS))
             THEN                                                   ; only one case
                  `[COND
                      ((DISCRIMINATE-2 ,MINCHARS ,(CAAR FORMS))
                       ,@(CDAR FORMS]
           ELSE                                                     ; Discriminate on the first code and recur on the tails
                (LIST* 'CASE `(CAR CODEVAR)
                       (WHILE FORMS BIND REST C
                          COLLECT (SETQ REST (CL:REMOVE (SETQ C (CAAAR FORMS))
```

```
                                                    FORMS :KEY 'CAAR))
                                  `(,C (SETQ CODEVAR (CDR CODEVAR))
                                       (DISCRIMINATE-1 ,(SUB1 MINCHARS)
                                              ,@(FOR F IN (CL:SET-DIFFERENCE FORMS (SETQ FORMS REST))
                                                   COLLECT (CONS (CDAR F)
                                                                 (CDR F]
(DISCRIMINATE-2 (MINCHARS MATCHLST)
```

;; True if codes match MATCHLST, with prefix at least MINCHARS long.

```
    (IF (NULL MATCHLST)
        THEN `(NULL CODEVAR)
      ELSE (LET [(CODE `(AND (EQ (CAR CODEVAR)
                                 ,(POP MATCHLST))
                             (PROGN (SETQ CODEVAR (CDR CODEVAR))
                                    (DISCRIMINATE-2 ,(SUB1 MINCHARS)
                                           ,MATCHLST]
                   (IF (<= MINCHARS 0)
                       THEN                                    ; Ok to match null
                            `(OR (NULL CODEVAR)
                                 ,CODE)
                     ELSE                                      ; Must match exactly so far
                          CODE]
(LET ((CODEVAR MONTH))                                         ; This LET is solely to allow more compact code (PVAR_ is one
                                                               ; byte less than IVARX_)
     (DISCRIMINATE '(("JANUARY" 1)
                     ("FEBRUARY" 2)
                     ("MARCH" 3)
                     ("APRIL" 4)
                     ("MAY" 5)
                     ("JUNE" 6)
                     ("JULY" 7)
                     ("AUGUST" 8)
                     ("SEPTEMBER" 9)
                     ("OCTOBER" 10)
                     ("NOVEMBER" 11)
                     ("DECEMBER" 12])
```

(\OUTDATE

```
[LAMBDA (UD FORMAT STRING)                                     ; Edited 30-May-89 12:28 by bvm
  (DESTRUCTURING-BIND
   (YEAR MONTH DAY HOUR MINUTE SECOND DST WDAY)
   UD
   (LET ((SEPR (CHARCODE -))
         (HOUR.LENGTH 2)
         SIZE S N NO.DATE NO.TIME NO.LEADING.SPACES TIME.ZONE TIME.ZONE.LENGTH YEAR.LENGTH MONTH.LENGTH
         DAY.LENGTH WDAY.LENGTH NO.SECONDS NUMBER.OF.MONTH MONTH.LONG MONTH.LEADING YEAR.LONG DAY.OF.WEEK
         DAY.SHORT CIVILIAN.TIME)
        (if (NOT FORMAT)
            then NIL
          elseif (NEQ (CAR (LISTP FORMAT))
                      'DATEFORMAT)
            then (LISPERROR "ILLEGAL ARG" FORMAT)
          else (for TOKEN in FORMAT
                  do (SELECTQ TOKEN
                             (NO.DATE (SETQ NO.DATE T))
                             (NO.TIME (SETQ NO.TIME T))
                             (NUMBER.OF.MONTH
                                  (SETQ NUMBER.OF.MONTH T))
                             (YEAR.LONG (SETQ YEAR.LONG T))
                             (MONTH.LONG (SETQ MONTH.LONG T))
                             (MONTH.LEADING
                                  (SETQ MONTH.LEADING T))
                             (SLASHES (SETQ SEPR (CHARCODE /)))
                             (SPACES (SETQ SEPR (CHARCODE SPACE)))
                             (NO.LEADING.SPACES
                                  (SETQ NO.LEADING.SPACES T))
                             (TIME.ZONE (SETQ TIME.ZONE (OR [LISTP (CDR (if (FIXP \TimeZoneComp)
                                                                            then (ASSOC \TimeZoneComp TIME.ZONES)
                                                                          else
                                                                ; Ugh, not a small integer
                                                                                 (CL:ASSOC \TimeZoneComp TIME.ZONES
                                                                                        :TEST '=]
                                                                       \TimeZoneComp)))
                             (NO.SECONDS (SETQ NO.SECONDS T))
                             (DAY.OF.WEEK (SETQ DAY.OF.WEEK T))
                             (DAY.SHORT (SETQ DAY.SHORT T))
                             (CIVILIAN.TIME
                                  (SETQ CIVILIAN.TIME T))
                            NIL)))
        [SETQ SIZE
         (+ (if NO.DATE
                then 0
              else (+ (if MONTH.LEADING
                          then (SETQ SEPR (CHARCODE SPACE))
                               (SETQ NUMBER.OF.MONTH NIL)       ; Will use a comma
                               1
```

```
                    else 0)
                   (SETQ MONTH.LENGTH
                    (if NUMBER.OF.MONTH
                        then                                        ; Month input is zero-based
                             (if (AND (< (add MONTH 1)
                                         10)
                                      NO.LEADING.SPACES)
                                 then 1
                                 else 2)
                          else [SETQ MONTH
                                (CL:NTH MONTH
                                       '("January" "February" "March" "April" "May" "June" "July" "August"
                                         "September" "October" "November" "December"]
                                (if MONTH.LONG
                                    then (NCHARS MONTH)
                                    else 3)))
                   (SETQ DAY.LENGTH (if (AND (OR NO.LEADING.SPACES MONTH.LEADING)
                                             (< DAY 10))
                                        then 1
                                        else 2))
                   (SETQ YEAR.LENGTH (if (OR YEAR.LONG (> YEAR 1999))
                                         then 4
                                         else (SETQ YEAR (IREMAINDER YEAR 100))
                                              2))
                   (if DAY.OF.WEEK
                       then [SETQ DAY.OF.WEEK (CL:NTH WDAY '("Monday" "Tuesday" "Wednesday" "Thursday"
                                                             "Friday" "Saturday" "Sunday"]
                            [+ 3 (SETQ WDAY.LENGTH (if DAY.SHORT
                                                       then ; 3 letters plus " ()"
                                                            3
                                                       else (NCHARS DAY.OF.WEEK]
                       else 0)
                   2))
          (if NO.TIME
              then 0
            else (+ (if NO.DATE
                        then 5
                        else 6)
                    (if NO.SECONDS
                        then 0
                        else 3)
                    (if CIVILIAN.TIME
                        then                                        ; Use AM/PM
                             (SETQ CIVILIAN.TIME (if (> HOUR 11)
                                                     then       ; PM
                                                          (if (> HOUR 12)
                                                              then (add HOUR -12))
                                                          (CHARCODE p)
                                                   else (if (EQ HOUR 0)
                                                            then (SETQ HOUR 12))
                                                        (CHARCODE a)))
                             (if (AND (< HOUR 10)
                                      NO.LEADING.SPACES)
                                 then (SETQ HOUR.LENGTH 1)
                                 else 2)
                        else 0)
                    (if (NULL TIME.ZONE)
                        then 0
                        elseif (NUMBERP TIME.ZONE)
                        then                                        ; Use the -0800 format
                             6
                        else                                        ; Depends on dst: (normal dst).  If missing, we are forced to use
                                                                    ; numeric format
                            (SETQ TIME.ZONE (OR (if DST
                                                    then (CADR TIME.ZONE)
                                                    else (CAR TIME.ZONE))
                                                \TimeZoneComp))
                            (ADD1 (SETQ TIME.ZONE.LENGTH (NCHARS TIME.ZONE]
    (SETQ S (ALLOCSTRING SIZE (CHARCODE SPACE)))
    (if (NOT NO.DATE)
        then (if MONTH.LEADING
                 then                                               ; Month day, year
                      (RPLSTRING S 1 MONTH)
                      (SETQ N MONTH.LENGTH)
                      (RPLCHARCODE S (add N 1)
                             SEPR)
                      (\RPLRIGHT S (add N (if (< DAY 10)
                                              then 1
                                              else 2))
                             DAY 1)
                      (RPLCHARCODE S (add N 1)
                             (CHARCODE ","))
               else                                                 ; Day<sepr>month<sepr>year
                    (\RPLRIGHT S (SETQ N DAY.LENGTH)
                           DAY 1)
                    (RPLCHARCODE S (add N 1)
                           SEPR)
```

```
                        (if NUMBER.OF.MONTH
                            then (\RPLRIGHT S (add N MONTH.LENGTH)
                                              MONTH MONTH.LENGTH)
                          else (\OUTDATE-STRING S N MONTH (NOT MONTH.LONG))
                               (add N MONTH.LENGTH)))
                    (RPLCHARCODE S (add N 1)
                           SEPR)
                    (\RPLRIGHT S (add N YEAR.LENGTH)
                           YEAR 2)
                    (OR NO.TIME (add N 1))
                    [if DAY.OF.WEEK
                        then                                    ; Day of week at very end in parens
                             (LET [(START (SUB1 (- SIZE WDAY.LENGTH]
                                  (RPLCHARCODE S START (CHARCODE "("))
                                  (\OUTDATE-STRING S START DAY.OF.WEEK DAY.SHORT)
                                  (RPLCHARCODE S SIZE (CHARCODE ")")]
      else (SETQ N 0))
    [if (NOT NO.TIME)
        then (\RPLRIGHT S (add N HOUR.LENGTH)
                    HOUR
                    (if CIVILIAN.TIME
                        then 1
                      else 2))
             (RPLCHARCODE S (ADD1 N)
                    (CHARCODE %:))
             (\RPLRIGHT S (add N 3)
                    MINUTE 2)
             (if (NOT NO.SECONDS)
                 then (RPLCHARCODE S (ADD1 N)
                             (CHARCODE %:))
                      (\RPLRIGHT S (add N 3)
                             SECOND 2))
             (if CIVILIAN.TIME
                 then (RPLCHARCODE S (ADD1 N)
                             CIVILIAN.TIME)
                      (RPLCHARCODE S (add N 2)
                             (CHARCODE m)))
             (if TIME.ZONE
                 then (if (NUMBERP TIME.ZONE)
                          then                                  ; +0800 etc
                               (if DST
                                   then                         ; Daylight savings is in effect, so time zone is off by an hour
                                        (SETQ TIME.ZONE (SUB1 TIME.ZONE)))
                               (RPLCHARCODE S (+ N 2)
                                      (if (<= TIME.ZONE 0)
                                          then                  ; East of GMT, which is denoted + in this notation
                                               (SETQ TIME.ZONE (- TIME.ZONE))
                                               (CHARCODE +)
                                        else (CHARCODE -)))
                               (if (FIXP TIME.ZONE)
                                   then                         ; integral number of hours
                                        (\RPLRIGHT S (+ N 4)
                                               TIME.ZONE 2)
                                        (RPLSTRING S (+ N 5)
                                               "00")
                                 else (CL:MULTIPLE-VALUE-BIND (H M)
                                        (CL:TRUNCATE TIME.ZONE)
                                        (\RPLRIGHT S (+ N 4)
                                               H 2)
                                        (\RPLRIGHT S (+ N 6)
                                               (ROUND (TIMES M 60))
                                               2)))
                            else (RPLSTRING S (+ N 2)
                                        TIME.ZONE]
    (if STRING
        then (SUBSTRING S 1 -1 STRING)
      else S])
```

(**OUTDATE-STRING**

```
  [LAMBDA (S N STRING SHORTP)                                   ; Edited 18-May-89 18:38 by bvm
```

;; Append STRING to S, using only the first 3 chars if SHORTP is true. N is the index of the last char appended to S. Returns new N

```
    (if SHORTP
        then                                                    ; Use only first 3 chars
             (for I from 1 to 3 do (RPLCHARCODE S (+ N I)
                                          (NTHCHARCODE STRING I)))
      else (RPLSTRING S (ADD1 N)
                  STRING])
```

(**UNPACKDATE**

```
  [LAMBDA (D)                                                   ; Edited  4-May-89 18:18 by bvm
```

;; Converts an internal Lisp date D into a list of integers (Year Month Day Hours Minutes Seconds daylightp DayOfWeek). D defaults to current ;; date. --- DayOfWeek is zero for Monday --- --- D is first converted to the alto standard, a 32-bit unsigned integer, representing the number of ;; seconds since jan 1, 1901-Gmt. We have to be a little tricky in our computations to avoid the sign bit.

```
    (SETQ D (OR D (DAYTIME)))
    (PROG ((CHECKDLS \DayLightSavings)
           (DQ (IQUOTIENT (LRSH (LISP.TO.ALTO.DATE D)
                                1)
                          30))
           MONTH SEC HR DAY4 YDAY WDAY YEAR4 TOTALDAYS MIN DLS FRAC)
                                                                  ; DQ is number of minutes since day 0, getting us past the sign
                                                                  ; bit problem.
          (SETQ SEC (IMOD [+ D (CONSTANT (- 60 (IMOD MIN.FIXP 60]
                          60))
          (SETQ MIN (IREMAINDER DQ 60))
```

;; Now we can adjust to the current time zone. Since this might cause DQ to go negative, first add in 4 years worth of hours, making the base ;; date be Jan 1, 1897

```
          [LET ((ZONE \TimeZoneComp))
               (if (NOT (FIXP ZONE))
                   then                                           ; Gack, a non-hour offset. Use the integer here, then adjust the
                                                                  ; minutes, etc.
                        (CL:MULTIPLE-VALUE-SETQ (ZONE FRAC)
                               (CL:FLOOR ZONE)))
               (SETQ HR (IREMAINDER (SETQ DQ (- (+ (IQUOTIENT DQ 60)
                                                   (CONSTANT (ITIMES 24 \4YearsDays)))
                                                ZONE))
                                    24))
               (if FRAC
                   then (SETQ FRAC (ROUND (TIMES FRAC -60)))      ; Minutes to add (time zones are never below the minute offset)
                        (CL:MULTIPLE-VALUE-SETQ (FRAC MIN)
                               (CL:FLOOR (+ MIN FRAC)
                                         60))
                        (if (NEQ FRAC 0)
                            then                                  ; Adjust the hours
                                 (CL:MULTIPLE-VALUE-SETQ (FRAC HR)
                                        (CL:FLOOR (+ HR FRAC)
                                                  24]
          (SETQ TOTALDAYS (IQUOTIENT DQ 24))
          (if FRAC
              then                                                ; For non-integral time zones, here's the last of the leftover.
                   (add TOTALDAYS FRAC))
      DTLOOP
          (SETQ DAY4 (IREMAINDER TOTALDAYS \4YearsDays))          ; DAY4 = number of days since last leap year day 0
          [SETQ DAY4 (+ DAY4 (CDR (\DTSCAN DAY4 '((789 . 3)
                                                  (424 . 2)
                                                  (59 . 1)
                                                  (0 . 0]         ; pretend every year is a leap year, adding one for days after Feb
                                                                  ; 28
          (SETQ YEAR4 (IQUOTIENT TOTALDAYS \4YearsDays))          ; YEAR4 = number of years til that last leap year / 4
          (SETQ YDAY (IREMAINDER DAY4 366))                       ; YDAY is the ordinal day in the year (jan 1 = zero)
          (SETQ WDAY (IREMAINDER (+ TOTALDAYS 3)
                                 7))
          (if (AND CHECKDLS (SETQ DLS (\ISDST? YDAY HR WDAY)))
              then ;; This date is during daylight savings, so add 1 hour. Third arg is day of the week, which we determine by taking days mod 7
                   ;; plus offset. Monday = zero in this scheme. Jan 1 1897 was actually a Friday (not Thursday=3), but we're cheating--1900
                   ;; was not a leap year
                   (if (> (SETQ HR (ADD1 HR))
                          23)
                       then ;; overflowed into the next day. This case is too hard (we might have overflowed the month, for example), so just go
                            ;; back and recompute
                            (SETQ TOTALDAYS (ADD1 TOTALDAYS))
                            (SETQ HR 0)
                            (SETQ CHECKDLS NIL)
                            (GO DTLOOP)))
          [SETQ MONTH (\DTSCAN YDAY '((335 . 11)
                                      (305 . 10)
                                      (274 . 9)
                                      (244 . 8)
                                      (213 . 7)
                                      (182 . 6)
                                      (152 . 5)
                                      (121 . 4)
                                      (91 . 3)
                                      (60 . 2)
                                      (31 . 1)
                                      (0 . 0]                     ; Now return year, month, day, hr, min, sec
          (RETURN (LIST (+ 1897 (ITIMES YEAR4 4)
                           (IQUOTIENT DAY4 366))
                        (CDR MONTH)
                        (ADD1 (- YDAY (CAR MONTH)))
                        HR MIN SEC DLS WDAY])

)

(RPAQQ TIME.ZONES
       ((8 "PST" "PDT")
        (7 "MST" "MDT")
        (6 "CST" "CDT")
```

```
        (5 "EST" "EDT")
        (0 "GMT" "BST")
        (0 "UT")
        (-1 "MET" "MET DST")
        (-2 "EET" "EET DST")))

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(RPAQQ \4YearsDays 1461)

(CONSTANTS \4YearsDays)
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS TIME.ZONES \TimeZoneComp \DayLightSavings)
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(SPECVARS *STR* *POS*)
)
)

(PUTPROPS DATEPATCH COPYRIGHT ("Venue & Xerox Corporation" 1989 1990))
```

FUNCTION INDEX

VARIABLE INDEX

CONSTANT INDEX