File created: 14-Jun-2024 14:54:24 **{WMEDLEY}<lispusers>QIX.;4**

edit by: rmk

changes to: (FNS QIX.GROW)

previous date: 14-Jun-2024 14:49:48 {WMEDLEY}<lispusers>QIX.;3

Read Table: XCL

Package: INTERLISP

Format: XCCS

```
(RPAQQ QIXCOMS ((FNS QIX.GROW QIX.IDLE QIX.MOVE.POINT QIX.PLAY)
                (RECORDS QIX.POINT)
                (P (SETQ IDLE.FUNCTIONS (CONS '("5 Qix's" 'QIX.IDLE)
                                              IDLE.FUNCTIONS)))))

(DEFINEQ
```

(QIX.GROW

```
  (LAMBDA (WINDOW DONTDISMISS)                                  ; Edited 14-Jun-2024 14:54 by rmk
                                                                ; Edited  1-Aug-87 16:57 by JEFF.SHRAGER
```

;;; This sets up a QIX the specified window. The QIX's parameters are defined at random, but with reasonable value ranges. The dismiss argument tell ;;; the QIX whether to DISMISS every cycle or not. Be careful.

```
    (PROG (P P2 (W (OR WINDOW (CREATEW)))
           L)
          (SETQ *STOP.QIXS* NIL)
```

;;; P and P2 define a QIX.

```
          (SETQ P (|create| QIX.POINT
                         QX _ (RAND 1 200)
                         QY _ (RAND 1 100)
                         VH _ (RAND 1 20)
                         VV _ (RAND 1 20)))
          (SETQ P2 (|create| QIX.POINT
                          QX _ (RAND 1 200)
                          QY _ (RAND 1 100)
                          VH _ (RAND 1 20)
                          VV _ (RAND 1 20)))
```

;;; L is the tail list. It starts out full of NILs and gets filled as the QIX moves. It is also inserted in it's own mouth so that the whole thing wraps around.

```
          (SETQ L (APPEND (|for| X |from| 1 |to| (RAND 5 25) |collect| (COPY '(A S D F)))
                          (LIST (LIST (|fetch| (QIX.POINT QX)
                                              P)
                                      (|fetch| (QIX.POINT QY)
                                              P)
                                      (|fetch| (QIX.POINT QX)
                                              P2)
                                      (|fetch| (QIX.POINT QY)
                                              P2)))))
          (RPLACD (LAST L)
                  L)
     LOOP
          (COND
             (*STOP.QIXS* (RPLACD L NIL)
                    (RETURN NIL)))
```

;;; Draw the QIX's head line.

```
          (MOVETO (|fetch| (QIX.POINT QX)
                           P)
                  (|fetch| (QIX.POINT QY)
                           P)
                  W)
          (DRAWTO (|fetch| (QIX.POINT QX)
                           P2)
                  (|fetch| (QIX.POINT QY)
                           P2)
                  1
                  'REPLACE W)
```

;;; Move the points according to their QX and QY velocities.

```
          (QIX.MOVE.POINT P W)
          (QIX.MOVE.POINT P2 W)
```

;;; Take a deep breath if the user asks you to. This slows things down.

```
          (OR DONTDISMISS (DISMISS))
```

;;; Delete the first object on the tail list.

```
          (COND
             ((EQ (CAAR L)
                  'A))
             (T (PROG ((OLD (CAR L)))
                      (MOVETO (CAR OLD)
                              (CADR OLD)
                              W)
                      (DRAWTO (CADDR OLD)
                              (CADDDR OLD)
                              1
                              'ERASE W))))
```

;;; Replace the current point with the new head, which effectively adds it to the end of the list, since we them immediately move to the next elt in this ;;; circular list.

```
          (RPLACA (CAR L)
                 (|fetch| (QIX.POINT QX)
                          P))
          (RPLACA (CDAR L)
                 (|fetch| (QIX.POINT QY)
                          P))
          (RPLACA (CDDAR L)
                 (|fetch| (QIX.POINT QX)
                          P2))
          (RPLACA (CDDDAR L)
                 (|fetch| (QIX.POINT QY)
                          P2))
          (SETQ L (CDR L))
          (GO LOOP))))
```

(QIX.IDLE

```
  (LAMBDA (W)                                          ; Edited 14-Jun-2024 14:49 by rmk
                                                       ; Edited 24-Aug-2022 07:53 by larry
                                                       ; Edited  1-Aug-87 16:58 by JEFF.SHRAGER
```

;;; CLOBBER ANY OLD QIXS THAT WERE LEFT AROUND (WASTING SPACE) FROM BEFORE.

```
    (AND (BOUNDP '*OLD-QIXS*)
         (FOR Q IN *OLD-QIXS* DO (RPLACD Q NIL)))
    (PROG (P P2 L QIXS)
```

;;; P and P2 define a QIX.

```
          (SETQ QIXS (|for| I |from| 1 |to| 5
                        |collect| (PROGN (SETQ P (|create| QIX.POINT
                                                           QX _ (RAND 1 200)
                                                           QY _ (RAND 1 100)
                                                           VH _ (RAND 1 20)
                                                           VV _ (RAND 1 20)))
                                         (SETQ P2 (|create| QIX.POINT
                                                            QX _ (RAND 1 200)
                                                            QY _ (RAND 1 100)
                                                            VH _ (RAND 1 20)
                                                            VV _ (RAND 1 20)))
```

;;; L is the tail list. It starts out full of NILs and gets filled as the QIX moves. It is also inserted in it's own mouth so that the whole thing wraps around.

```
                                         (SETQ L (APPEND (|for| X |from| 1 |to| (RAND 5 25)
                                                            |collect| (COPY '(A S D F)))
                                                         (LIST (LIST (|fetch| (QIX.POINT QX)
                                                                              P)
                                                                     (|fetch| (QIX.POINT QY)
                                                                              P)
                                                                     (|fetch| (QIX.POINT QX)
                                                                              P2)
                                                                     (|fetch| (QIX.POINT QY)
                                                                              P2)))))
                                         (RPLACD (LAST L)
                                                 L)
                                         (LIST P P2 L))))
          (SETQ *OLD-QIXS* QIXS)
      LOOP
          (BLOCK 25)
          (|for| Q |in| QIXS |do| (SETQ P (CAR Q))
                                  (SETQ P2 (CADR Q))
                                  (SETQ L (CADDR Q))
```

;;; Draw the QIX's head line.

```
                                  (MOVETO (|fetch| (QIX.POINT QX)
                                                   P)
                                          (|fetch| (QIX.POINT QY)
                                                   P)
                                          W)
                                  (DRAWTO (|fetch| (QIX.POINT QX)
                                                   P2)
```

```
                                        (|fetch| (QIX.POINT QY)
                                               P2)
                                        1
                                        'REPLACE W)
```

;;; Move the points according to their QX and QY velocities.

```
                                    (QIX.MOVE.POINT P W)
                                    (QIX.MOVE.POINT P2 W)
```

;;; Delete the first object on the tail list.

```
                                    (COND
                                       ((EQ (CAAR L)
                                            'A))
                                       (T (PROG ((OLD (CAR L)))
                                                (MOVETO (CAR OLD)
                                                       (CADR OLD)
                                                       W)
                                                (DRAWTO (CADDR OLD)
                                                       (CADDDR OLD)
                                                       1
                                                       'ERASE W))))
```

;;; Replace the current point with the new head, which effectively adds it to the end of the list, since we THEN immediately move to the next elt in this ;;; circular list.

```
                                    (RPLACA (CAR L)
                                           (|fetch| (QIX.POINT QX)
                                                  P))
                                    (RPLACA (CDAR L)
                                           (|fetch| (QIX.POINT QY)
                                                  P))
                                    (RPLACA (CDDAR L)
                                           (|fetch| (QIX.POINT QX)
                                                  P2))
                                    (RPLACA (CDDDAR L)
                                           (|fetch| (QIX.POINT QY)
                                                  P2))
                                    (RPLACA (CDDR Q)
                                           (CDR L)))
            (GO LOOP))))
```

(QIX.MOVE.POINT

```
  (LAMBDA (P W)                                                    ; Edited 14-Jun-2024 14:48 by rmk
                                                                   (* |edited:| "16-May-85 00:39")
```

;;; This guy updates the QIX line endpoints according to their velocities in the X and Y directions. If we hit a wall, then simply negate the relevant velocity ;;; vector.

```
    (PROG ((VV (|fetch| VV P))
           (VH (|fetch| VH P))
           (X (|fetch| (QIX.POINT QX)
                     P))
           (Y (|fetch| (QIX.POINT QY)
                     P)))
          (PROG ((NEWX (IPLUS X VH))
                 (NEWY (IPLUS Y VV)))
                (COND
                   ((LESSP NEWY 0)
                    (SETQ NEWY 0)
                    (SETQ VV (ITIMES -1 VV)))
                   ((GREATERP NEWY (WINDOWPROP W 'HEIGHT))
                    (SETQ NEWY (WINDOWPROP W 'HEIGHT))
                    (SETQ VV (ITIMES -1 VV))))
                (COND
                   ((LESSP NEWX 0)
                    (SETQ NEWX 0)
                    (SETQ VH (ITIMES -1 VH)))
                   ((GREATERP NEWX (WINDOWPROP W 'WIDTH))
                    (SETQ NEWX (WINDOWPROP W 'WIDTH))
                    (SETQ VH (ITIMES -1 VH))))
                (|replace| (QIX.POINT QY)
                       P NEWY)
                (|replace| (QIX.POINT QX)
                       P NEWX)
                (|replace| VV P VV)
                (|replace| VH P VH)))))
```

(QIX.PLAY

```
  (LAMBDA (N)                                                      (* |Jeff.Shrager| " 8-Sep-85 14:01")
                                                                   (* "Jeff Shrager" "24-May-84 22:17")
```

(* |This| |takes| |over| |the| |screen| |and| |sets| |up| \a |number| |of| QIX |on| |it.|
I\t |also| |hangs| |itself| |at| |the| |end| |so| |that| |the| TTY |window| |doesn't| |come| |to| |the| |surface.|)

```
    (PROG ((W (CREATEW '(0 0 1024 830)
                       NIL 1)))
          (|for| X |from| 1 |to| N |do| (ADD.PROCESS (LIST 'QIX.GROW (KWOTE W))))
          (UNTILMOUSESTATE (AND LEFT RIGHT MIDDLE)))))

)

(DECLARE\: EVAL@COMPILE

(RECORD QIX.POINT (QX QY VH VV))
)

(SETQ IDLE.FUNCTIONS (CONS '("5 Qix's" 'QIX.IDLE)
                           IDLE.FUNCTIONS))
```

FUNCTION INDEX

RECORD INDEX
