CROCK

By: Kelly Roach

New Owner: Herb Jellinek (Jellinek.pa@Xerox.com)

CROCK sets up an analog face clock in the user's environment. To use, LOAD CROCK.LCOM and call (CROCK). CROCK requires that PROCESSWORLD be running (automatic in Fugue or later).

CROCK

Function CROCK has the form

(CROCK *REGION*) [Function]

The first invocation creates a clock window, CROCKWINDOW, occupying REGION with style CROCK.DEFAULT.STYLE. If REGION is left NIL, a region will be prompted for. Subsequent invocations use CROCKWINDOW. Only one clock window may exist at any given time. The clock is updated once a minute.

STYLE

The clock's style is maintained as a property list and can be found by (WINDOWPROP CROCKWINDOW 'STYLE). There are four independent boolean properties which the user may control: HANDS (the hands of the clock), TIMES (time digits printed where the hands end), RINGS (rings on the clock face), and NUMBERS (12 numbers around the outside of the clock face). The style first used will be CROCK.DEFAULT.STYLE (bound to '(HANDS T TIMES NIL RINGS NIL NUMBERS T) when CROCK is first loaded).

CROCK.DATEFORMAT

The user can control how the date will be printed in CROCKWINDOW. CROCK.DATEFORMAT should have the form (DATEFORMAT . <tokens>) where each <token> is one of NO.DATE, NO.TIME, NUMBER.OF.MONTH, YEAR.LONG, SLASHES, SPACES, NO.LEADING.SPACES, TIME.ZONE, or NO.SECONDS. These are all listed on pp23.57-58 of the IRM. Unfortunately, some other possibilities, such as DAY.OF.WEEK have not been implemented by Interlisp-D yet and are therefore not available to CROCK yet. The default value for CROCK.DATEFORMAT is (DATEFORMAT NO.SECONDS). For example,

```
(SETQ CROCK.DATEFORMAT
    '(DATEFORMAT SLASHES NUMBER.OF.MONTH NO.SECONDS))
```

would make CROCK print a date string like

```
28/09/84 14:53
```

instead of a date string like

```
28-Sep-84 14:53
```

Since CROCK updates itself only once a minute, it is probably a good idea to always include NO.SECONDS in your CROCK.DATEFORMAT.

CROCK.ALARM AND CROCK.TUNE

The user can set CROCK's alarm via

(CROCK.ALARM *DATESTRING*) [Function]

where DATESTRING is any arg acceptable to Interlisp's IDATE (such as the date CROCK prints in CROCKWINDOW). CROCK will act appropriately when time reaches DATESTRING. Dandelion users can set global CROCK.TUNE to a tune to be played by Interlisp's PLAYTUNE when CROCK's alarm acts.

RECOMMENDED USAGE

The simplest way to call CROCK from your init file or other function is to set your CROCK globals, then call CROCK:

(SETQ CROCK.DEFAULT.STYLE *STYLE*) [Variable]

(SETQ CROCK.DATEFORMAT *DATEFORMAT*) [Variable]

(SETQ CROCK.TUNE *TUNE*) [Variable]

(CROCK *REGION*) [Function]

You supply <style>, <dateformat>, <tune>, and <region>. You only need the SETQs if you want non-default values. If no <region> is supplied, CROCK will prompt for one.

LEFT MOUSE BUTTON

Buttoning CROCKWINDOW with the left mouse button requests immediate update of the clock. (Of course, it may take a while for the process scheduler to get to it.)

MIDDLE MOUSE BUTTON

Buttoning CROCKWINDOW with the middle mouse button presents a menu of commands for modifying the clock's style. Menu item SHOW.STYLE prints the clock's style.

RIGHT MOUSE BUTTON

Buttoning CROCKWINDOW with the left mouse button presents the usual window menu. RESHAPEing the CROCKWINDOW causes the clock to change its size to fit the new window region. CLOSEing the CROCKWINDOW deletes the clock process.