

File created: 5-Dec-2020 16:35:40 {DSK}<Users>arunwelch>SKYDRIVE>DOCUMENTS>UNIX>LISP>LDE>ROOM
S>MEDLEY-35>ROOMS-BACKGROUNDS.;2

previous date: 17-Aug-90 12:29:44 {DSK}<Users>arunwelch>SKYDRIVE>DOCUMENTS>UNIX>LISP>LDE>ROOMS>MEDLEY-35>ROOMS-B
ACKGROUNDS.;1

Read Table: XCL

Package: ROOMS

Format: XCCS

; Copyright (c) 1987, 1988, 1990, 2020 by Venue & Xerox Corporation. All rights reserved.

```
(IL:RPAQQ IL:ROOMS-BACKGROUNDS
  ;; code for painting background
  (FILE-ENVIRONMENTS IL:ROOMS-BACKGROUNDS)
  (IL:P (EXPORT ' (MAKE-BACKGROUND BACKGROUND-INTERNAL-FORM *DEFAULT-BACKGROUND-TEXT-FONT*
    RENAISSANCE-BITMAP SQUARES-BITMAP TILE-BITMAP INTERNALIZE-ALL-BACKGROUNDS))
    (REQUIRE "ROOMS"))
  (IL:STRUCTURES BACKGROUND)
  (IL:FUNCTIONS MAKE-BACKGROUND INTERNALIZE-BACKGROUND INTERNALIZE-BACKGROUND-TEXT
    INTERNALIZE-ALL-BACKGROUNDS)
  (EVAL-WHEN (LOAD)
    (IL:P (PUSHNEW 'INTERNALIZE-ALL-BACKGROUNDS *SCREEN-CHANGED-FUNCTIONS*)))
  (IL:VARIABLES *DEFAULT-BACKGROUND* *DEFAULT-BACKGROUND-TEXT-FONT*)
  (IL:FUNCTIONS PAINT-BACKGROUND FIND-BACKGROUNDS DRAW&FILL-BOX-WITHIN)
  (IL:VARIABLES *SCREEN-BITMAP*)
  (IL:VARIABLES RENAISSANCE-BITMAP SQUARES-BITMAP TILE-BITMAP)
  (IL:GLOBALVARS IL:WINDOWBACKGROUNDSHADE IL:WHOLESCREEN)))

;; code for painting background

(DEFINE-FILE-ENVIRONMENT IL:ROOMS-BACKGROUNDS :COMPILER :COMPILE-FILE
  :PACKAGE "ROOMS"
  :READTABLE "XCL")

(EXPORT ' (MAKE-BACKGROUND BACKGROUND-INTERNAL-FORM *DEFAULT-BACKGROUND-TEXT-FONT* RENAISSANCE-BITMAP
  SQUARES-BITMAP TILE-BITMAP INTERNALIZE-ALL-BACKGROUNDS))

(REQUIRE "ROOMS")

(DEFSTRUCT (BACKGROUND (:CONSTRUCTOR MAKE-BACKGROUND-INTERNAL)
  (EXTERNAL-FORM NIL :TYPE LIST :READ-ONLY T)
  ;; what the user sees
  (INTERNAL-FORM NIL :TYPE LIST)
  ;; what PAINT-BACKGROUND operates on
  )

(DEFUN MAKE-BACKGROUND (EXTERNAL-FORM)
  (MAKE-BACKGROUND-INTERNAL :EXTERNAL-FORM EXTERNAL-FORM :INTERNAL-FORM (INTERNALIZE-BACKGROUND EXTERNAL-FORM
  )))
```

```
(DEFUN INTERNALIZE-BACKGROUND (BACKGROUND)
  ;; internalize BACKGROUND
  (MAPCAR #' (LAMBDA (PROP-LIST)
    (CASE (CAR PROP-LIST)
      (:WHOLE-SCREEN
        (CHECK-TYPE (SECOND PROP-LIST)
          (OR BITMAP TEXTURE))
        (CHECK-TYPE (GETF PROP-LIST :BORDER)
          (OR NULL INTEGER))
        (CHECK-TYPE (GETF PROP-LIST :BORDER-SHADE)
          (OR NULL TEXTURE))
        ` (:WHOLE-SCREEN , (SECOND PROP-LIST)
          , @ (WHEN (GETF PROP-LIST :BORDER)
            ` (:BORDER , (GETF PROP-LIST :BORDER)))
          , @ (WHEN (GETF PROP-LIST :BORDER-SHADE)
            ` (:BORDER-SHADE , (GETF PROP-LIST :BORDER-SHADE))))))
      (:REGION
        (CHECK-TYPE (GETF PROP-LIST :SHADE)
          (OR NULL BITMAP TEXTURE))
        (CHECK-TYPE (GETF PROP-LIST :BORDER)
          (OR NULL INTEGER))
        (CHECK-TYPE (GETF PROP-LIST :BORDER-SHADE)
          (OR NULL TEXTURE))
        ` (:REGION , (INTERNALIZE-REGION (SECOND PROP-LIST))
          , @ (WHEN (GETF PROP-LIST :SHADE)
            ` (:SHADE , (GETF PROP-LIST :SHADE)))
```

```

,@ (WHEN (GETF PROP-LIST :BORDER)
      `(:BORDER ,(GETF PROP-LIST :BORDER)))
,@ (WHEN (GETF PROP-LIST :BORDER-SHADE)
      `(:BORDER-SHADE ,(GETF PROP-LIST :BORDER-SHADE))))
(:TEXT (LIST :TEXT (LET ((*DEFAULT-TEXT-FONT* *DEFAULT-BACKGROUND-TEXT-FONT*))
                        (INTERNALIZE-BACKGROUND-TEXT PROP-LIST))))
(EVAL-WALK BACKGROUND)))

```

```

(DEFUN INTERNALIZE-BACKGROUND-TEXT (PROP-LIST)
  (MAKE-TEXT :STRING (GETF PROP-LIST :TEXT)
    :POSITION
    (INTERNALIZE-POSITION (OR (GETF PROP-LIST :POSITION)
                              (MAKE-POSITION 0 0)))
    :ALIGNMENT
    (OR (GETF PROP-LIST :ALIGNMENT)
        :LEFT-BOTTOM)
    :FONT
    (LET ((FONT (GETF PROP-LIST :FONT)))
      (IF FONT
        (IL:FONTCREATE FONT)
        *DEFAULT-TEXT-FONT*))
    :SHADOWS
    (GETF PROP-LIST :SHADOWS T)))

```

```

(DEFUN INTERNALIZE-ALL-BACKGROUNDS ()
  ;; do all the named rooms
  (DO-ROOMS (ROOM)
    (LET ((BACKGROUND (ROOM-BACKGROUND ROOM)))
      (SETF (BACKGROUND-INTERNAL-FORM BACKGROUND)
            (INTERNALIZE-BACKGROUND (BACKGROUND-EXTERNAL-FORM BACKGROUND))))))
  ;; do the Overview too (yes, this is ugly)
  (LET ((BACKGROUND (ROOM-BACKGROUND *OVERVIEW-ROOM*))
        (SETF (BACKGROUND-INTERNAL-FORM BACKGROUND)
              (INTERNALIZE-BACKGROUND (BACKGROUND-EXTERNAL-FORM BACKGROUND))))
    NIL)

```

```
(EVAL-WHEN (LOAD)
```

```
(PUSHNEW 'INTERNALIZE-ALL-BACKGROUNDS *SCREEN-CHANGED-FUNCTIONS*)
)
```

```
(DEFGLOBALPARAMETER *DEFAULT-BACKGROUND* (MAKE-BACKGROUND `((:WHOLE-SCREEN ,IL:WINDOWBACKGROUNDSHADE)
)))
```

```
(DEFPARAMETER *DEFAULT-BACKGROUND-TEXT-FONT* (IL:FONTCREATE 'IL:TIMESROMAND 36))
```

```

(DEFUN PAINT-BACKGROUND (ROOM DSP &KEY (SCALE *ONE-TO-ONE*)
                             NO-TEXT CLIPPING-REGION)
  (DOLIST (BACKGROUND (FIND-BACKGROUNDS ROOM))
    (DOLIST (SPEC (BACKGROUND-INTERNAL-FORM BACKGROUND))
      (CASE (FIRST SPEC)
        (:WHOLE-SCREEN (DRAW&FILL-BOX-WITHIN (SCALE-REGION IL:WHOLESCREEN SCALE)
                                              DSP :SHADE (SECOND SPEC)
                                              :BORDER-WIDTH
                                              (SCALE-WIDTH (GETF SPEC :BORDER 0)
                                                            SCALE)
                                              :BORDER-SHADE
                                              (GETF SPEC :BORDER-SHADE IL:BLACKSHADE)
                                              :CLIPPING-REGION CLIPPING-REGION))
          (:REGION (DRAW&FILL-BOX-WITHIN (SCALE-REGION (GETF SPEC :REGION)
                                                       SCALE)
                                          DSP :SHADE (GETF SPEC :SHADE)
                                          :BORDER-WIDTH
                                          (SCALE-WIDTH (GETF SPEC :BORDER 0)
                                                            SCALE)
                                          :BORDER-SHADE
                                          (GETF SPEC :BORDER-SHADE IL:BLACKSHADE)
                                          :CLIPPING-REGION CLIPPING-REGION))
            (:TEXT (UNLESS NO-TEXT
                      (DISPLAY-TEXT (SECOND SPEC)
                                     DSP :SCALE SCALE))))))

```

```
(DEFUN FIND-BACKGROUNDS (ROOM)
```

;;; returns the list of backgrounds which apply to ROOM

```

(LET (BACKGROUNDS FOUND-WHOLE-SCREEN?)
  (DO-INCLUSIONS (ROOM ROOM)
    (LET ((BACKGROUND (ROOM-BACKGROUND ROOM)))

```

```

(PUSH BACKGROUND BACKGROUNDS)
;; stop when we see one which paints the whole background
(WHEN (ASSOC :WHOLE-SCREEN BACKGROUND :TEST 'EQ)
      (SETQ FOUND-WHOLE-SCREEN? T)
      (RETURN-FROM DO-INCLUSIONS)))
(UNLESS FOUND-WHOLE-SCREEN? (PUSH *DEFAULT-BACKGROUND* BACKGROUNDS))
BACKGROUNDS))

```

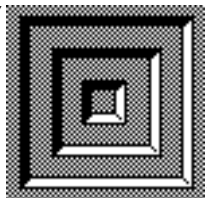
```

(DEFUN DRAW&FILL-BOX-WITHIN (REGION DSP &KEY (SHADE IL:WHITESHAE)
                                (BORDER-WIDTH 0)
                                CLIPPING-REGION
                                (BORDER-SHADE IL:BLACKSHAE))
  (LET ((LEFT (REGION-LEFT REGION))
        (BOTTOM (REGION-BOTTOM REGION))
        (WIDTH (REGION-WIDTH REGION))
        (HEIGHT (REGION-HEIGHT REGION)))
    (IF (OR (NULL BORDER-WIDTH)
            (ZEROP BORDER-WIDTH))
        (PAINT-REGION DSP REGION SHADE CLIPPING-REGION)
        (LET ((TOP (+ BOTTOM HEIGHT))
              (RIGHT (+ LEFT WIDTH))
              (INSIDE-LEFT (+ LEFT BORDER-WIDTH))
              (INSIDE-WIDTH (- WIDTH BORDER-WIDTH)))
            (PAINT-REGION DSP (MAKE-REGION :LEFT INSIDE-LEFT :BOTTOM (+ BOTTOM BORDER-WIDTH)
                                           :WIDTH INSIDE-WIDTH :HEIGHT (- HEIGHT BORDER-WIDTH))
                          SHADE CLIPPING-REGION)
            ;; up left
            (IL:BLTSHADE BORDER-SHADE DSP LEFT BOTTOM BORDER-WIDTH HEIGHT BORDER-WIDTH NIL CLIPPING-REGION)
            ;; across top
            (IL:BLTSHADE BORDER-SHADE DSP INSIDE-LEFT (- TOP BORDER-WIDTH)
                          INSIDE-WIDTH BORDER-WIDTH NIL CLIPPING-REGION)
            ;; up the right
            (IL:BLTSHADE BORDER-SHADE DSP (- RIGHT BORDER-WIDTH)
                          BOTTOM BORDER-WIDTH (- HEIGHT BORDER-WIDTH)
                          NIL CLIPPING-REGION)
            ;; across the bottom
            (IL:BLTSHADE BORDER-SHADE DSP INSIDE-LEFT BOTTOM INSIDE-WIDTH BORDER-WIDTH NIL CLIPPING-REGION)
            )))

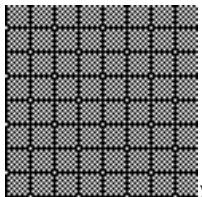
```

```
(DEFGLOBALVAR *SCREEN-BITMAP* (IL:SCREENBITMAP))
```

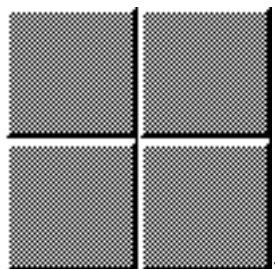
```
(DEFGLOBALVAR RENAISSANCE-BITMAP '
```



```
(DEFGLOBALPARAMETER SQUARES-BITMAP '
```



```
(DEFGLOBALPARAMETER TILE-BITMAP '
```



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

{MEDLEY}<rooms>ROOMS-BACKGROUNDS.;1

(IL:GLOBALVARS IL:WINDOWBACKGROUNDSHADE IL:WHOLESCREEN)
)

(IL:PUTPROPS **IL:ROOMS-BACKGROUNDS IL:COPYRIGHT** ("Venue & Xerox Corporation" 1987 1988 1990 2020))

FUNCTION INDEX

DRAW&FILL-BOX-WITHIN	3	INTERNALIZE-BACKGROUND	1	PAINTE-BACKGROUND	2
FIND-BACKGROUNDS	2	INTERNALIZE-BACKGROUND-TEXT	2		
INTERNALIZE-ALL-BACKGROUNDS	2	MAKE-BACKGROUND	1		

VARIABLE INDEX

DEFAULT-BACKGROUND	2	*SCREEN-BITMAP*	3	SQUARES-BITMAP	3
DEFAULT-BACKGROUND-TEXT-FONT	2	RENAISSANCE-BITMAP	3	TILE-BITMAP	3

STRUCTURE INDEX

BACKGROUND	1
------------------	---

FILE-ENVIRONMENT INDEX

IL:ROOMS-BACKGROUNDS	1
----------------------------	---
