EditBitMap provides an interface (EDIT.BITMAP) for creating and editing bitmaps, which may exist as named files or as part of another type of a file (for example, a document written in TEdit).

EditBitMap puts up a menu of bitmap-manipulation commands, one of which is HAND.EDIT, which accesses EDITBM, the Interlisp-D bitmap editor.

EditBitMap also works on cursors (produces new cursor) and symbols (works on the value and resets the value with the result).

Requirements

READNUMBER
SCALEBITMAP

Installation

Load EDITBITMAP.LCOM and the required .LCOM modules from the library.

User Interface

The user interface consists of a function (EDIT.BITMAP), a main operation menu, and a three-part window for low-level pixel editing.

There are two principal ways of entering the bitmap editor. If the bitmap is an object in a document being edited, you can enter the bitmap editor by pressing the left button over the bitmap. If the bitmap is an object you are manipulating as part of a program, you can call the function EDIT.BITMAP from the Executive, passing it the bitmap (typically the value of some variable).

In either case, EditBitMap presents its main menu, from which you select the operation you desire. If the operation is "Hand Edit", EditBitMap brings up a three-part window to show, create, or edit a bitmap. The individual EditBitMap operations can also be performed programmatically or from the Executive (see "Functions").

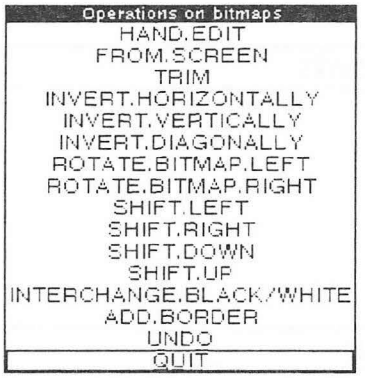
EDIT.BITMAP

The function EDIT.BITMAP is the principal way to create, view or edit bitmaps stored as the values of variables (or other easily accessible Lisp values):

(EDIT.BITMAP *BITMAP*) [Function]

BITMAP may be a bitmap, a cursor, or a symbol. If *BITMAP* is a bit map, then EDIT.BITMAP returns a new bitmap as the result of the edit. If *BITMAP* is a cursor, then EDIT.BITMAP operates on its bitmap and returns a new cursor. If *BITMAP* is a symbol, then EDIT.BITMAP operates on the symbol's value (a bitmap or cursor), and resets the symbol's value to the result of the edit.

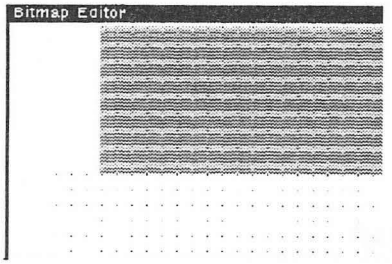
EditBitMap brings up a main menu containing the following items:



EditBitMap performs each command you select, until you select QUIT, at which point it returns the final result of all the edits. You can select UNDO to undo the most recent operation (selecting it several times undoes several operations). If you select HAND.EDIT, you enter the pixel editor EDITBM (see the *IRM*). If you select FROM.SCREEN, EditBitMap prompts you for a screen region from which to initialize a new bit map. The remaining menu items are described under the corresponding "Function" below.

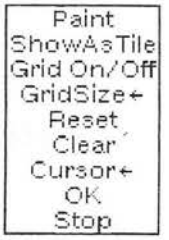
Window

The EditBitMap window consists of three parts. The main, lower part is the region where the bitmap is displayed on a grid by means of fat pixels. The smaller top part is a gray background against which the middle-button submenu is displayed. The small portion in the upper left corner displays a miniature picture of the entire bitmap.



Submenu

The EditBitMap submenu is displayed when you press the middle button in the upper gray region of the window. It contains the following items:

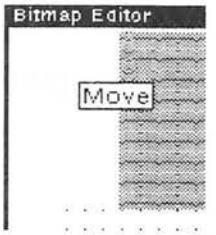


These menu items are described in the system prompt window when an item is selected.

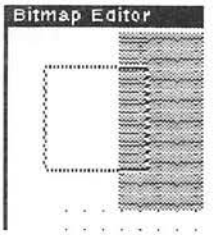
Mouse Buttons

Pressing the right button anywhere in the EditBitMap window causes the usual window menu to be displayed.

Pressing the left or middle button in the upper left portion of the window presents a MOVE icon.



Pressing on the MOVE icon presents a rectangle which can be moved about in the subwindow. This rectangle indicates the portion of the entire bitmap which will be displayed in the large, bottom subwindow, as soon as the button is released.



Pressing the middle button in the upper, gray subwindow causes the EditBitMap submenu to be displayed (see above).

Pressing the left button in the upper, gray subwindow highlights a rectangle in the upper left subwindow, showing which portion of the entire bitmap is displayed in the large lower subwindow.

Pressing the left button (or dragging the mouse with the left button held down) in the large, lower portion of the window changes the pixels; you are editing at the pixel level.

Editing an Existing Bitmap

If you have a variable *OLDNAME* whose value is a bitmap, you can make a modified version assigned to *NEWNAME* by typing to the Executive window:

```
(SETQ NEWNAME (EDIT.BITMAP OLDNAME))
```

Or if you want to modify *OLDNAME* in place, pass the quoted name itself:

```
(EDIT.BITMAP 'OLDNAME)
```

In either case, the main menu pops up on the screen. Select the operations you wish to perform, including HAND.EDIT to edit at the pixel level.

Edit the bitmap as needed.

Move the cursor into the gray upper region. Press the middle button to get the submenu. Select OK.

In the main menu, select QUIT. The Executive window displays the new bitmap address.

Viewing an Existing Bitmap

You can use the hand editor simply to view a bitmap. In this case you don't need to include a SETQ to save the value. For example, type

```
(EDIT.BITMAP BITMAPNAME)
```

Note: Any edits you might be tempted to make while viewing the bitmap in this way will not be saved.

Creating a New Bitmap

You can use any of the standard graphics interfaces documented in the *IRM* to create a new bitmap. EditBitMap does provide one convenient way to create a bitmap from a region of the screen. In the Executive window, type

```
(SETQ NEWBITMAPNAME (EDIT.BITMAP))
```

Again the main menu pops up on the screen. Select FROM.SCREEN. The cursor changes into the standard region prompt, allowing you to select a region of the screen. Hold down the left mouse button to mark one corner of the region, and drag the mouse to the opposite corner. When you let go of the mouse button, the contents of the screen region you selected are used to initialize a new bitmap. You can then select any other operations you wish to transform this initial image, including HAND.EDIT if appropriate. When finished with all the operations, select QUIT from the main menu. The variable *NEWBITMAPNAME* is now set to the bitmap you have created.

If you want to create a bitmap completely from scratch using the pixel editor, it is simplest to call it directly:

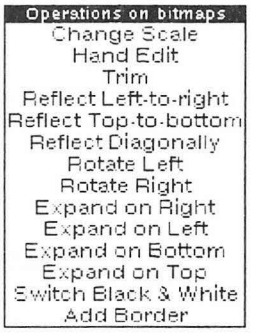
```
(SETQ NEWBITMAPNAME (EDITBM))
```

You will be prompted to supply the width and height of the new bitmap in pixels. When you are finished editing, move the cursor into the gray upper region, press the middle button to get the submenu, and select OK.

If you want to perform further transformations on the new bitmap, edit *NEWBITMAPNAME* as described above for existing bitmaps.

Editing a Bitmap in a Document

To edit a bitmap that exists inside another file, such as a document being edited in TEdit, press the left or middle button anywhere inside the image of the bitmap. A modified version of EditBitMap's main menu pops up containing the following items:



These menu items correspond exactly to the similarly-named items in EditBitMap's regular menu, except that only one operation is performed at a time (to repeat an operation, just select it again with the mouse; to Undo, use TEdit's Undo command). The menu also contains an additional item, CHANGE SCALE, which allows you to change the scale at which the bitmap's image appears in the document (on the screen and on the printer). Scaling a bitmap changes only the size of its image; it has no effect on its contents (even though on the screen you may not always see it that way).

Functions

(EDIT.BITMAP *BITMAP*) [Function]

(See above)

(ADD.BORDER.TO.BITMAP *BITMAP NBITS TEXTURE*) [Function]

Returns a new bitmap that is *BITMAP* extended by *NBITS* in all four directions, the border being filled in with *TEXTURE*.

(BIT.IN.COLUMN *BITMAP COLUMN*) [Function]

Returns T if any bit in column numbered *COLUMN* (left = 0) is not 0, NIL otherwise.

(BIT.IN.ROW *BITMAP ROW*) [Function]

Returns T if any bit in row numbered *ROW* (bottom = 0) is not zero, NIL otherwise.

(INVERT.BITMAP.B/W *BITMAP*) [Function]

Returns a new bitmap, which is *BITMAP* with all its bits inverted (black for white).

(INVERT.BITMAP.DIAGONALLY *BITMAP*) [Function]

Returns a new bitmap, which is *BITMAP* flipped about the X = Y diagonal. (The resulting bitmap's width will be *BITMAP*'s height.)

(INVERT.BITMAP.HORIZONTALLY *BITMAP*) [Function]

Returns a new bitmap, which is *BITMAP* flipped about its vertical center line.

(INVERT.BITMAP.VERTICALLY *BITMAP*) [Function]

Returns a new bitmap, which is *BITMAP* flipped about its horizontal center line.

(ROTATE.BITMAP.LEFT *BITMAP*) [Function]

Returns a new bitmap, which is *BITMAP* rotated 90 degrees counterclockwise. (The resulting bitmap's width will be *BITMAP*'s height.)

(ROTATE.BITMAP.RIGHT *BITMAP*) [Function]

Returns a new bitmap, which is BITMAP rotated 90 degrees clockwise. (The resulting bitmap's width will be *BITMAP*'s height.)

(SHIFT.BITMAP.DOWN *BITMAP NBITS*) [Function]

Returns a new bitmap, which is *BITMAP* extended by *NBITS* in the upward direction, the new space being filled in with white.

(SHIFT.BITMAP.UP *BITMAP NBITS*) [Function]

Returns a new bitmap, which is *BITMAP* extended by *NBITS* in the downwards direction, the new space being filled in with white.

(SHIFT.BITMAP.LEFT *BITMAP NBITS*) [Function]

Returns a new bitmap, which is *BITMAP* extended by *NBITS* to the right, the new space being filled in with white.

(SHIFT.BITMAP.RIGHT *BITMAP NBITS*) [Function]

Returns a new bitmap, which is *BITMAP* extended by *NBITS* to the left, the new space being filled in with white.

(TRIM.BITMAP *BITMAP*) [Function]

Returns a new bitmap, which is *BITMAP* trimmed at all four edges of all completely white (0) columns and rows.

(FROM.SCREEN.BITMAP NIL) [Function]

Prompts for a region on the screen and returns a copy of the bitmap.

(INTERACT&SHIFT.BITMAP.LEFT *BITMAP*) [Function]

Prompts for number of bits to shift the *BITMAP* left and returns the new bitmap.

(INTERACT&SHIFT.BITMAP.RIGHT *BITMAP*) [Function]

Prompts for number of bits to shift the *BITMAP* right and returns the new bitmap.

(INTERACT&SHIFT.BITMAP.DOWN *BITMAP*) [Function]

Prompts for number of bits to shift the *BITMAP* down and returns the new bitmap.

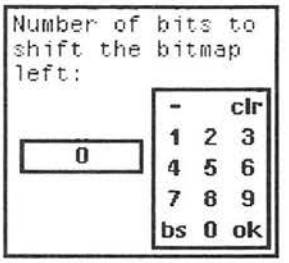
(INTERACT&SHIFT.BITMAP.UP *BITMAP*) [Function]

Prompts for number of bits to shift the *BITMAP* up and returns the new bitmap.

(INTERACT&ADD.BORDER.TO.BITMAP *BITMAP*) [Function]

Prompts for number of bits in the border and calls EDITSHADE to interactively fill in the texture. Returns a new bitmap, which is a bitmap extended in all four directions by the border being filled in with the texture.

Note: If the interactive functions are called from the menus, the prompt for the number of bits is in the form of a ReadNumber window:



Limitations

Selecting OK in the submenu does NOT save the edits made in the bitmap. Edits are saved only if you specify a new bitmap name before you begin editing an old one, or if you pass a quoted name to EDIT.BITMAP.