

EDM

Extensible Display Manager for EPICS

John Sinclair

June 25, 2001

Outline

- Introduction
- Program Execution
- Display File Commands
- Creating and Editing Objects
- Manipulating Objects on the Display
- Display Object Classes

Outline (cont)

- Display Execution
- Color Rules
- Macro Expansion
- Symbols
- Current List of Objects

Introduction

- EDM is an interactive GUI builder and execution engine, EPICS documentation uses the term *Display Manager*
- Maintained by ORNL EPICS community
- Component based, thus extensible by other members of the EPICS collaboration

Program Execution - Environment Variables

■ EDMDATAFILES

– export EDMDATAFILES=projectdir:userdir

e.g. export EDMDATAFILES=
/usr/displays:/home/me/mydisplays

Program Execution - Command Line Options

- Define symbols

- m “symbol1=value1,symbol2=value2,...”
(referenced as \$(symbol1))

- Execute mode

- x (-noedit)

- edm -x -noedit -m “unit=1” displayFile

Main Window Display File Operations

- New

- user directory

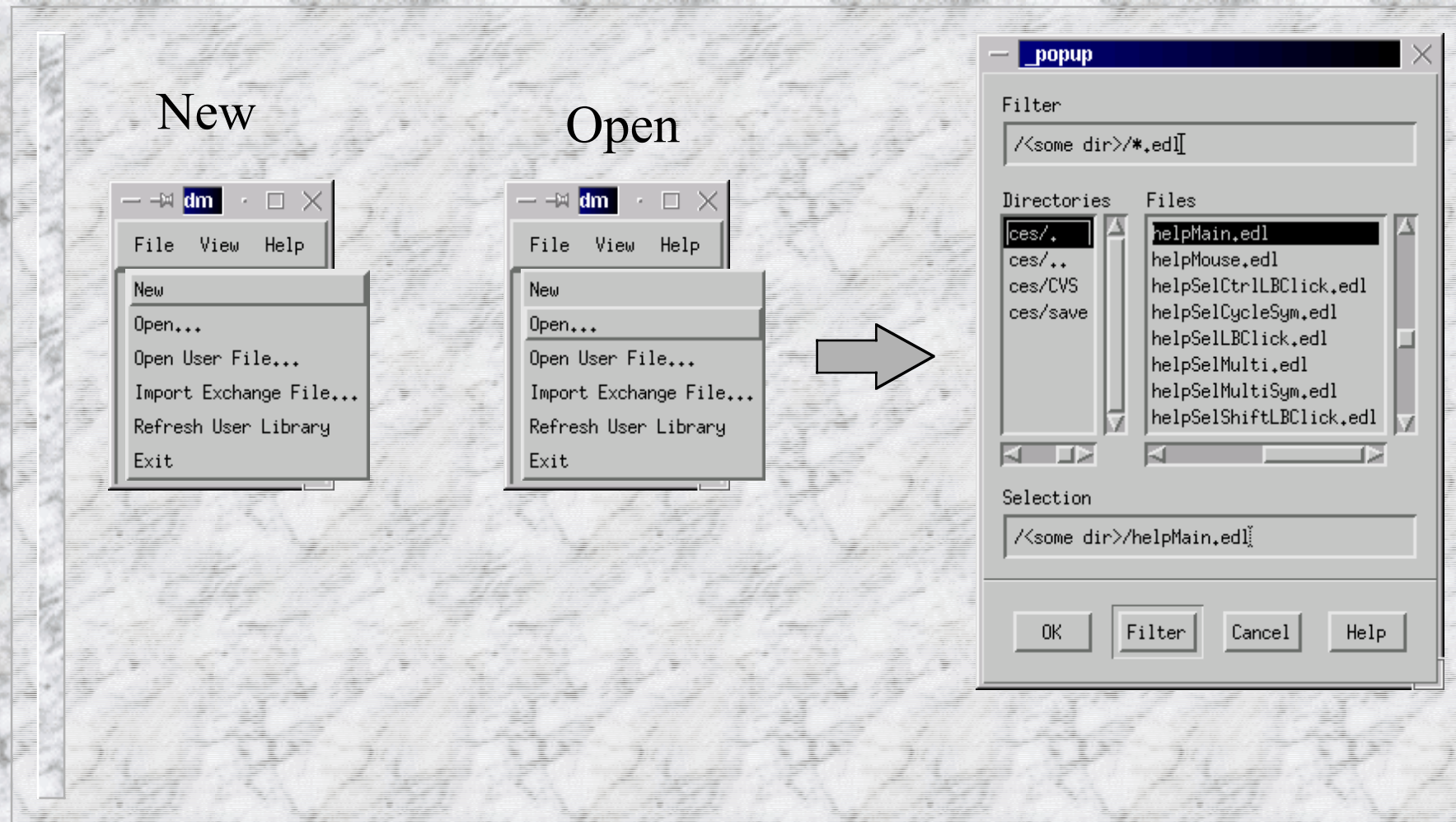
- Open

- project directory

- Open User File

- user directory

Main Window File Operations

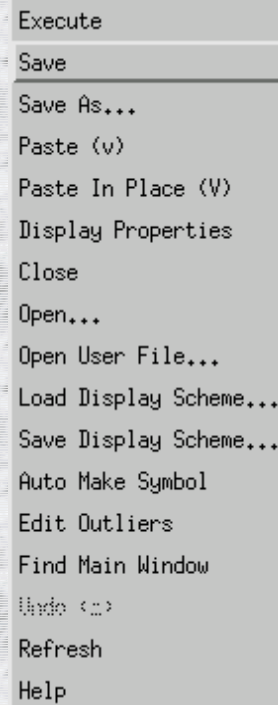


Display Menu File Operations

With no objects selected,
click the middle mouse
button on the display
background



This menu pops-up



Save
Save As...
Close
Open...
Open User File...

File Operation Notes

- You never need include the file extension in a file open or save operation
- “Save As...” to an existing file requires user confirmation

New Project...

- Set EDMDATAFILES
- Create a new display
- Edit display properties and set default fonts and colors
- Save display scheme as default.scheme
- Exit EDM

Creating Objects

The image illustrates the process of creating a circle object in a software application, shown in three sequential steps:

- Step 1:** A context menu is displayed over a canvas. The menu items are: Graphics, Monitors, Controls, Lines, Rectangle, Circle, Arc, Static Text, Scale, Archive X-Y Plot, GIF Image, PNG Image, and Dynamic Symbol. The 'Circle' option is highlighted. An arrow points from the text 'Left mouse button drag' to the start of the drag operation.
- Step 2:** The 'Circle Properties' dialog box is open, showing the following settings:
 - X: 202
 - Y: 497
 - Width: 50
 - Height: 50
 - Line Thk: 1
 - Line Style: Solid
 - Line Color: 79
 - Alarm Sensitive
 - Fill
 - Fill Color: grey-4
 - Alarm Sensitive
 - Color PV: [empty]
 - Visibility PV: [empty]
 - Visible if: [empty]
 - >=: [empty]
 - and <: [empty]Buttons at the bottom include OK, Apply, and Cancel.
- Step 3:** The final result is a circle object drawn on the canvas.

Creating Lines

The image illustrates the process of creating a line in a software application. It is divided into three main sections:

- Left Section:** A vertical toolbar on the left contains a 'Graphics' icon. An arrow labeled 'Left mouse button drag' points from this icon to a small rectangle drawn on the canvas.
- Middle Section:** A context menu is open over the rectangle, listing options: Graphics, Monitors, Controls, Lines, Rectangle, Circle, Arc, Static Text, Scale, Archive X-Y Plot, GIF Image, PNG Image, and Dynamic Symbol. A large grey arrow points from this menu to the right.
- Right Section:** A 'Lines Properties' dialog box is open, titled 'TopLevelShell'. It contains the following fields:
 - X: 45
 - Y: 48
 - Width: 129
 - Height: 182
 - Line Thk: 1
 - Line Style: Solid
 - Line Color: Red (with 'Monitor: MAJOR' text)
 - Fill Color: Grey (with 'Mid-alt/Anno-sec' text)
 - Color PV: [empty]
 - Visibility PV: [empty]
 - Visible if: [empty]
 - >=: [empty]
 - and <: [empty]At the bottom are 'OK', 'Apply', and 'Cancel' buttons. A large grey arrow points from the dialog box to the right.

On the far right, red lines and text indicate mouse actions:

- 'left click' points to the top-right corner of the rectangle.
- 'click' points to the bottom-right corner of the rectangle.
- 'click' points to the bottom-left corner of the rectangle.
- 'shift-click or double click' points to the bottom-right corner of the rectangle.

Selecting Objects

■ Left button click

- Single exclusive select: object is selected, currently selected objects are deselected

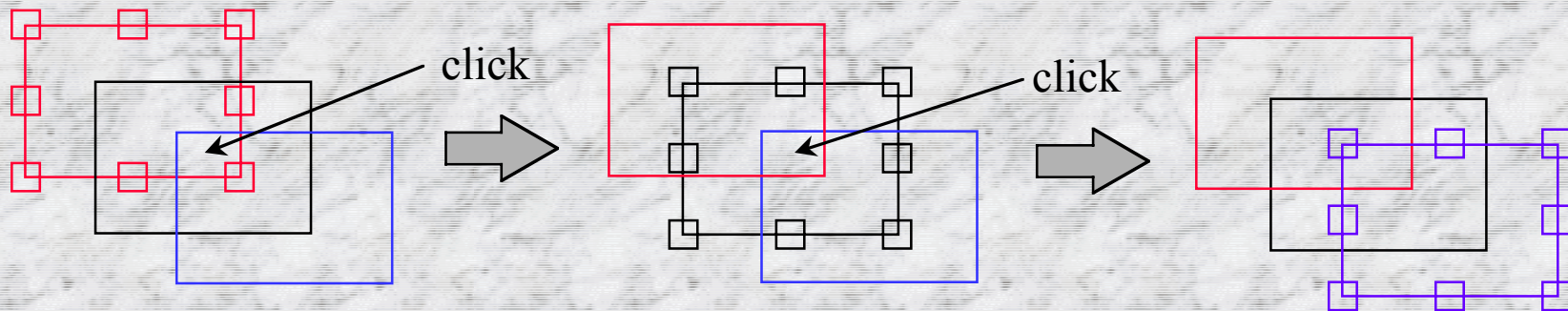
■ Shift-left button click

- Single inclusive select: object is added or removed from the current group of selected objects

Selecting Objects (cont)

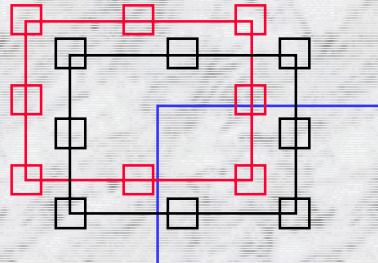
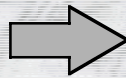
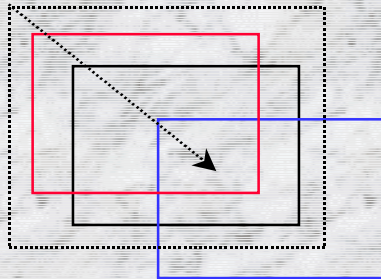
■ Control-left button click

- If only one object is currently selected then selection cycles among overlapping objects

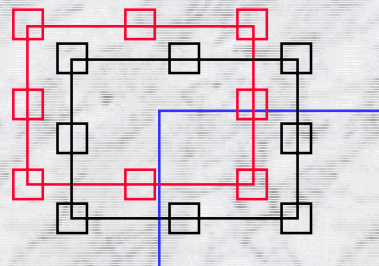
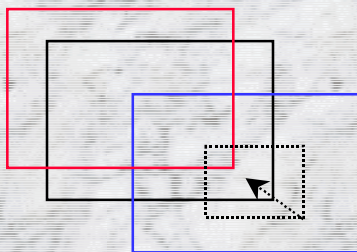


Selecting Objects (cont)

- Middle button drag - objects are added or removed from the current selection group



Top-left to bottom-right:
Select enclosed objects



Bottom-right to top-left:
Select enclosed corners

Editing Objects

left click on selected object

TopLevelShell

Rectangle Properties

X 316

Y 150

Width 57

Height 42

Line Thk 1

Line Style Solid

Line Color ■ Monitor: MAJOR

Alarm Sensitive

Fill

Fill Color ■ Mid-alt/Anno-sec

Alarm Sensitive

Invisible

Color PV :

Visibility PV :

Visible if

>= :

and < :

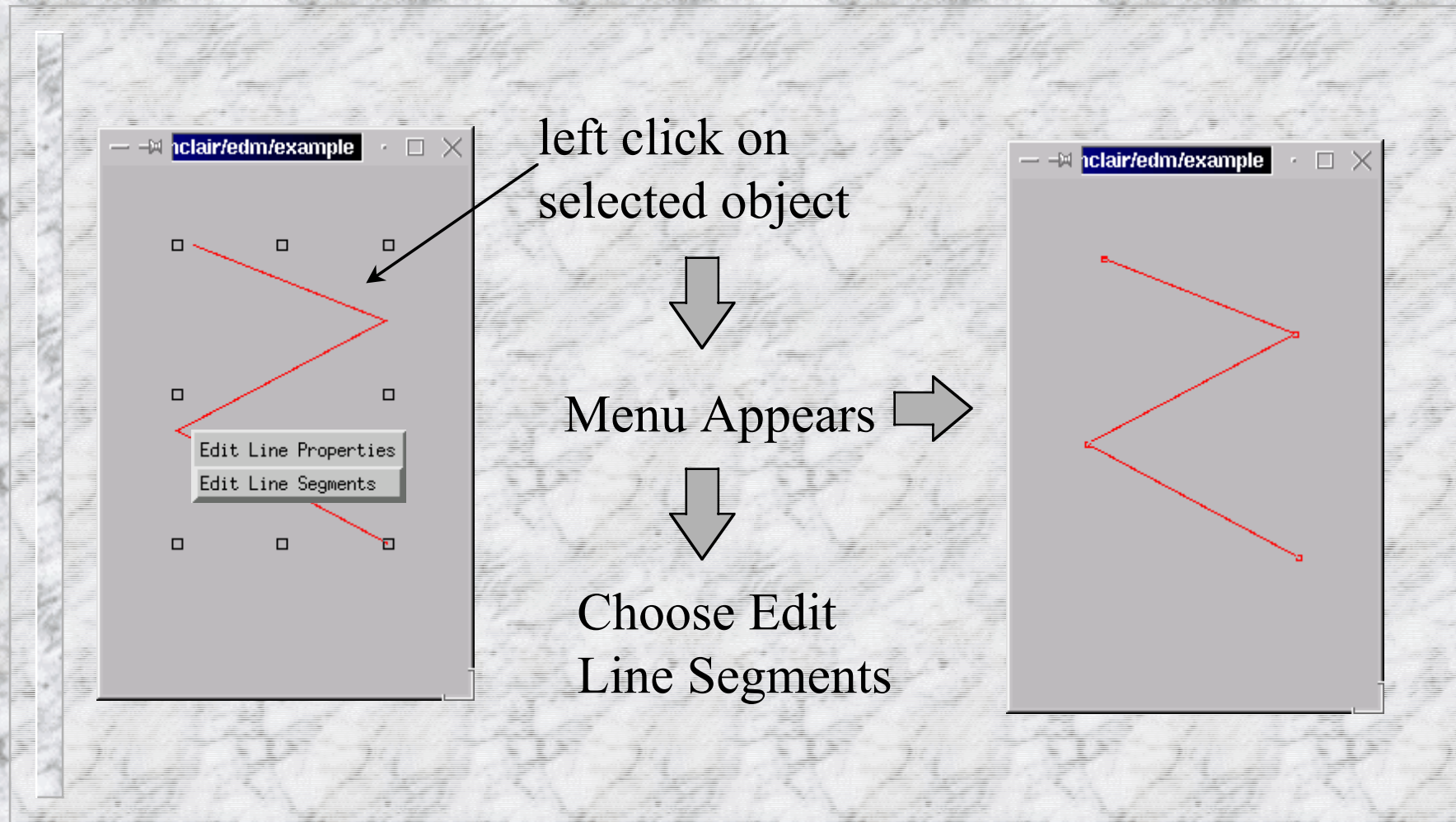
OK Apply Cancel

Editing Line Properties

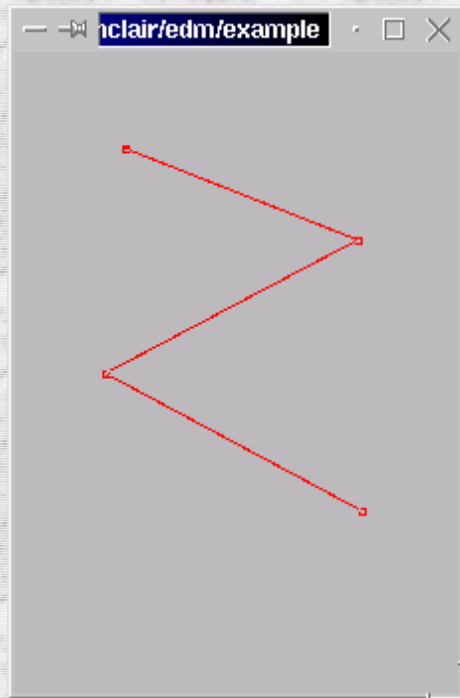
The diagram illustrates the steps to edit line properties in a software application. It consists of three main parts:

- Left Panel:** A screenshot of a window titled "Inclair/edm/example". It shows a red line connecting several square nodes. A context menu is open over the line, with options "Edit Line Properties" and "Edit Line Segments". An arrow points to the line with the text "left click on selected object".
- Flow Diagram:** A vertical sequence of three downward-pointing arrows. The text "Menu Appears" is positioned between the first and second arrows, and "Choose Edit Line Properties" is positioned between the second and third arrows. A horizontal arrow points from the "Menu Appears" text to the right.
- Right Panel:** A screenshot of a dialog box titled "TopLevelShell" with a sub-header "Lines Properties". It contains various input fields and checkboxes:
 - X: 45
 - Y: 48
 - Width: 129
 - Height: 182
 - Line Thk: 1
 - Line Style: Solid
 - Line Color: Red (with a color swatch and "Monitor: MAJOR" label)
 - Alarm Sensitive
 - Fill
 - Fill Color: Grey (with a color swatch and "Wid-alt/Anno-sec" label)
 - Alarm Sensitive
 - Color PV: [text field]
 - Visibility PV: [text field]
 - Visible if: [dropdown menu]
 - >=: [text field]
 - and <: [text field]Buttons for "OK", "Apply", and "Cancel" are at the bottom.

Editing Line Segments



Editing Line Segments (cont)

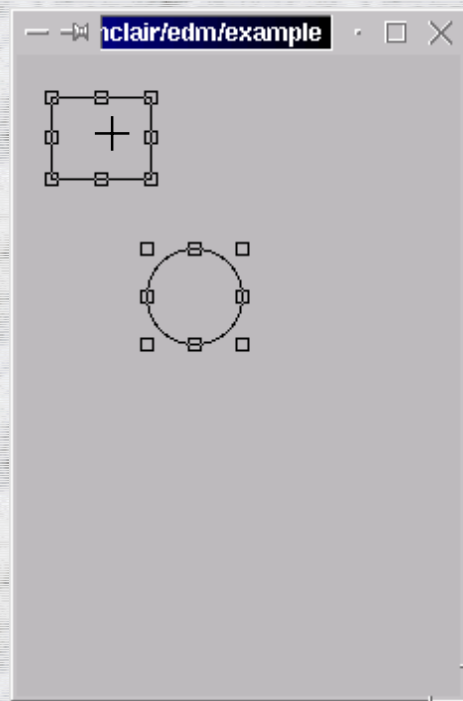


Left-click	Add point
Shift-middle-click	Delete last point
Middle-drag	Move point
Shift-left-click or Left-double-click	Terminate edit operation

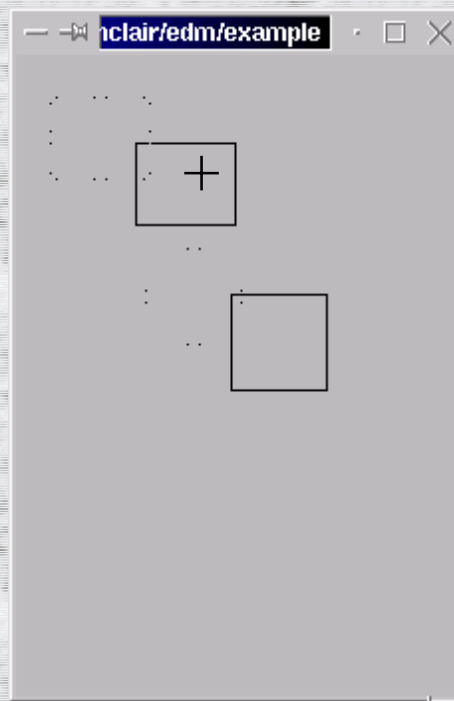
Editing Notes

- Clicking on one of a group of selected objects brings up the property box for each object, one-by-one, as the OK button is pressed.
- To minimize mouse movement, instead of clicking OK, Apply, or Cancel, you may double-click the left, middle, or right button respectively.

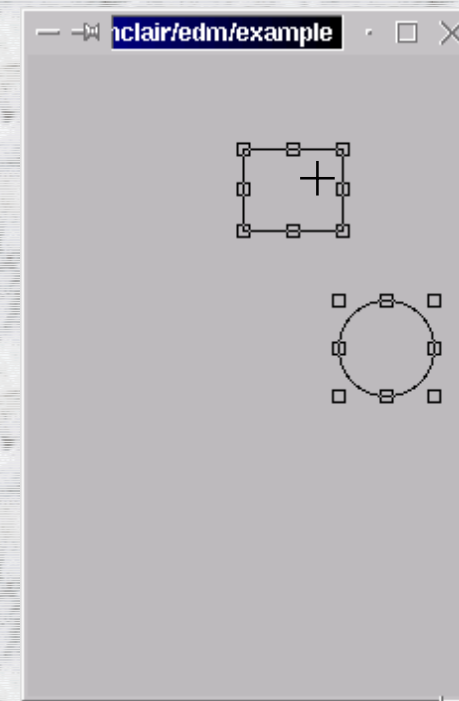
Moving Objects



Place mouse cursor
on interior of one
object

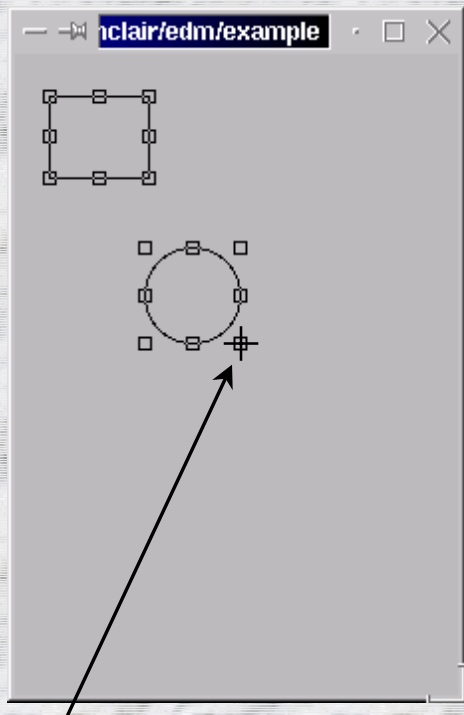


Press left button and
drag objects to new
location

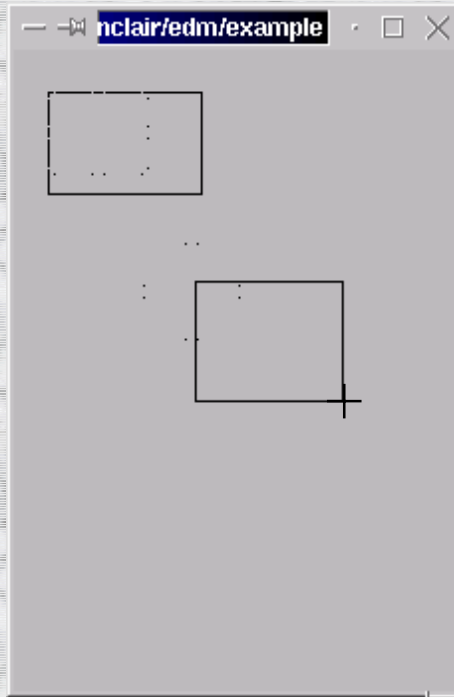


Release mouse
button

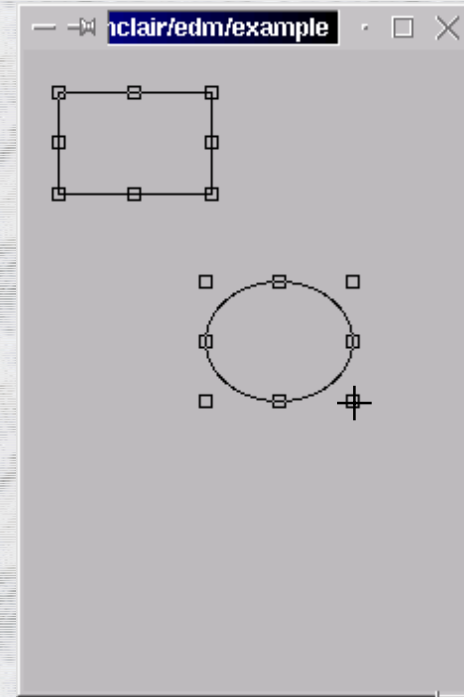
Resizing Objects



Place mouse cursor
on control point
of one object



Press left button and
drag to new size



Release mouse
button

Draw/Move/Resize Notes

- Fine control may be achieved on moves and resizes by using keyboard arrow keys (mouse button release or click ends op)
- Control key forces move (prevents resize)
- M/m key turns ON/off orthogonal move
- L/l key turns ON/off orthogonal line draw
- G/g key turns ON/off grid
- S/s key turns ON/off snap-to-grid

Alignment Operations

■ Reference Independent

- Align left, right, top, bottom
- Distribute: vert axis, horiz axis
- Distribute Midpoint: vert axis, horiz axis

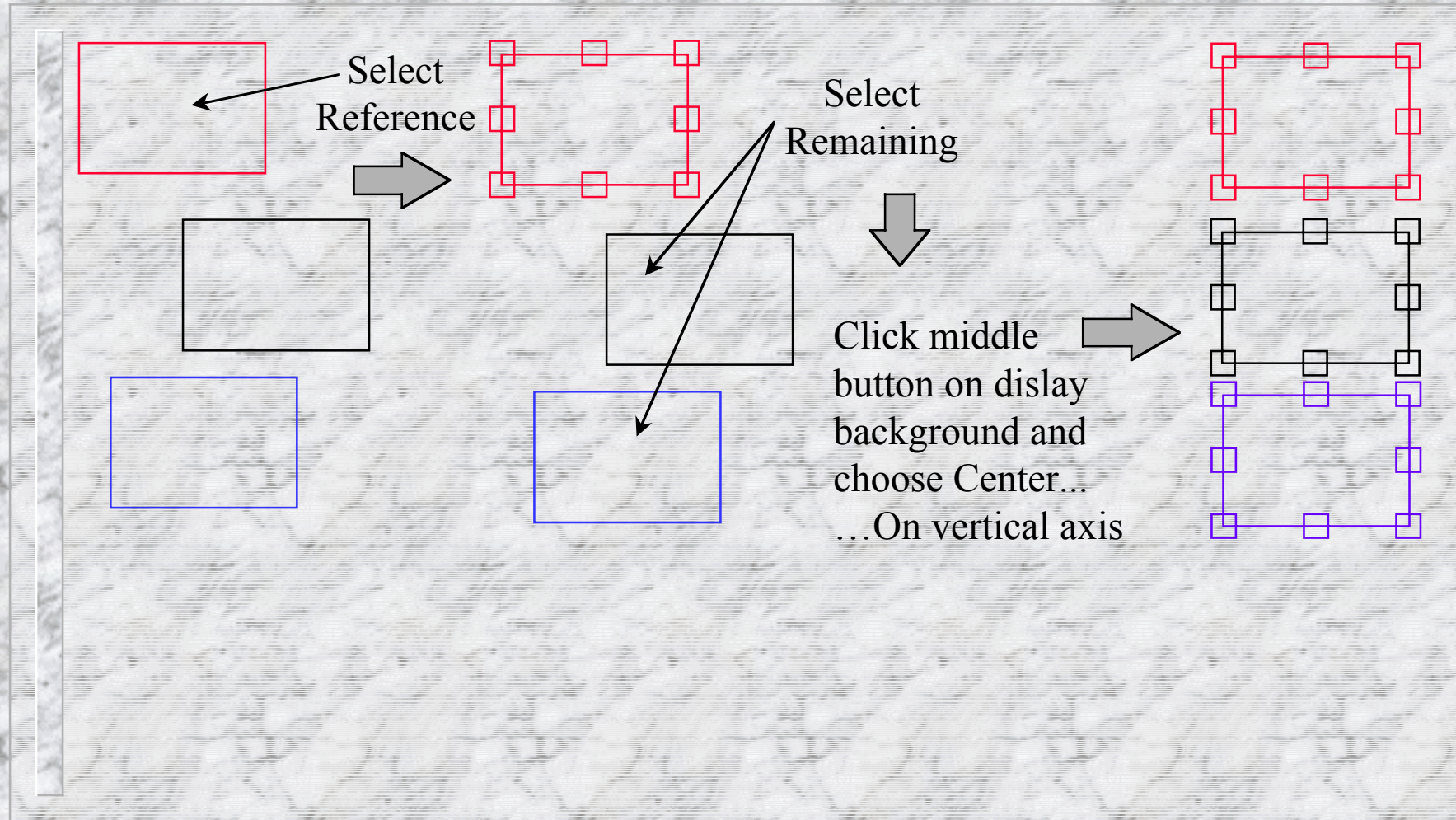
■ Reference Dependent

- Center: horizontal, vertical, both
- Size: width, height, both

Reference Dependent Operations

- First object selected is used as reference
- If no reference object is specified, an appropriate object is chosen (topmost, leftmost, etc.)

Example Align Operation



Misc Operations

- Raise, Lower
- Copy, Cut, Paste
- Group, Ungroup
- Flip H & V
- Rotate CW & CCW
- Group Edit
- Undo

Group Edit

- Change visual attributes of all selected objects
- Change PV names for all selected objects

Undo - Current Limitations

- Most useful for move, resize, & alignment operations
- Cannot undo edit operations
- Cut, Group, and Ungroup : Flush undo stack

Exercise 1- Display Schemes

- Execute edm
- Create a new display
- Invoke the middle mouse button menu to edit display attributes
- Select default fonts and colors, Click OK
- Invoke the menu again, select *Save Display Scheme*, make the file name *default.scheme*
- From the main window, choose file-->exit

Exercise 2

- Execute edm
- Create several graphic objects
- Explore the basic object manipulation operations
 - Select, copy, paste, move, resize, align, ...
- Explore the edit operation
 - Change colors, line thickness, ...
- Explore snap-to-grid, ortho move & line draw
- Save the display as example1

EDM Objects

■ Graphics

- Lines, rectangle, circle, arc, text, gif, png, dynamic symbol

■ Monitors

- Meter, bar, message box, symbol, text update, strip chart

■ Controls

- Text, slider, button, menu button, message button, up-down button, multiplexor, related display, shell command



Demo Screen

A semi-circular gauge with a scale from 0 to 100. The needle points to approximately 65.3.

orib36:ana1	1.0	65.3
orib36:ana2		76.4

0 save rest 100

A horizontal progress bar with a scale from 0 to 100. The bar is filled to approximately 65.3.

65.3 76.4

Disabled On On

EXIT

Display Execution

- Macro symbols are expanded
- Colors rules become active
- Visibility rules become active
- The rendering of control and monitor objects may change substantially; in execute mode certain components of these objects usually change in response to PV value changes and/or operator interaction

Execute-mode Behavior

- Color Rules
- Macro Expansion
- Illustration through several objects
 - Related Display
 - Shell Command
 - Exit Button
 - Static Text
 - Control Text

Specifying Color

The image displays three windows from a software application:

- TopLevelShell - Rectangle Properties:** A dialog box for configuring a rectangle. It includes fields for X (316), Y (150), Width (57), and Height (42). The Line Color is set to red (Monitor: MAJOR) and the Fill Color is set to a light gray (Mid-alt/Anno-sec). There are checkboxes for Alarm Sensitive, Fill, and Invisible, and a Visible if dropdown.
- Color:** A list of color names including test1, test1Inv, Disconn/Invalid, Wid-bc/Anno-pri, Wid-alt/Anno-sec, Monitor: MAJOR, Mon: MAJOR/unack, Monitor: MINOR, Mon: MINOR/unack, Monitor: NORMAL, Monitor: alt, Controller, Controller/alt, Related display, Shell/reldsp-alt, Exit/Quit/Kill, GLOBAL canvas, GLOBAL title, GLOBAL help, and FE canvas. The 'Wid-alt/Anno-sec' entry is highlighted.
- Color:** A color palette window showing a grid of color swatches. A red swatch is selected, corresponding to the 'Monitor: MAJOR' color in the 'TopLevelShell' window.

Arrows indicate the flow of information: from the 'Color' list to the 'Line Color' and 'Fill Color' fields in the 'TopLevelShell' window, and from the 'Color' list to the 'Color' palette.

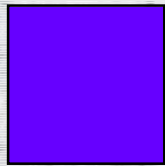
Color may be specified visually or by name

Color - Static and Dynamic

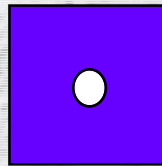
- Some color entries are dynamic and are associated with a color rule
- In execute mode, dynamic colors change as a function of the color rule operating on the current value of an associated EPICS PV
- An invisible color may be defined

Color - Static and Dynamic

- Colors may be specified for various object attributes and appear as one or more buttons in object property dialog boxes. Dynamic colors are differentiated from static colors in the following manner:



Static



Dynamic

Color Rules

- Color Rules are give in the edm color list file. The following is an example of a rule named *Red-or-Blue*:

```
Red-or-Blue rule {  
  <5      : Red  
  >=5     : Blue  
}
```

Using this rule, a rectangle object could be made to change color at run-time depending on the value of some EPICS pv

EDM Macro Expansion

- Macro symbol sources

- Command line
- Related Display parameter
- Multiplexor

- At run-time, symbol expands to associated value

e.g. -m “one=1” at run-time, \$(one) \longrightarrow 1

Exercise 3 (1 of 3)

■ Objective:

- Examine operation of dynamic colors
- Explore macro symbol expansion

■ Execute edm and open example1

■ Add a related display button

- 1st Entry, Filename: relatedDsp1, Symbols:param=1
- 2nd Entry, Filename: relatedDsp1, Symbols:param=2

■ Add a text control

- Editable=Yes, obtain the Control PV name from an instructor, this same PV name will be used in a color rule

■ Save the display file

Exercise 3 (2 of 3)

- Create a new display
 - Create a static text object with *Text Value* set to $\text{param}=\$(\text{param})$
 - Create a rectangle, choose a dynamic color for *line color*, obtain the color rule name from an instructor and use the PV name from above
 - Create an exit button
- Save the new display as *relatedDsp1* and close the display window
- Return to the example1 display, deselect all objects, click the middle mouse button, and choose *execute* from the menu

Exercise 3 (3 of 3)

- Click the Related Display button and choose a menu item, the associated display should appear and the static text object should display the symbol value
- Change the value of the PV from the example1 text control, the rectangle color should be determined by the color rule
- Experiment with various options
- When you have finished experimenting, click the Exit Button on each related display
- Exit edm

Symbols

- EDM implements a primitive symbol facility
- Symbols are multi-state objects where each state maps to a value range of an associated EPICS PV
- 64 states max, color and size may be changed per symbol instance if so desired

Symbols (cont)

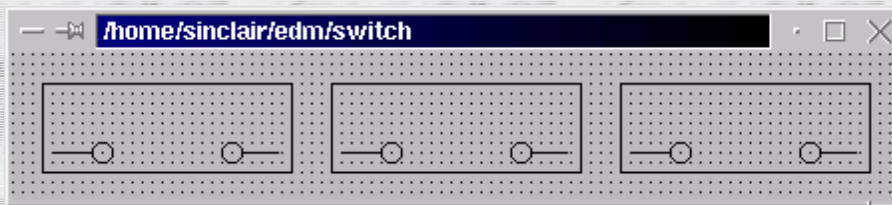
- An EDM symbol is nothing more than a standard display file where each symbol state is represented as a group of objects
- Only one grouping level is allowed
- The visual ordering corresponds to the ordering of states
- EDM contains an auto-make symbol command to perform the grouping and ordering

Creating Symbols

1. Create a rectangle corresponding to the geometric boundaries of the symbol, check the invisible attribute of this rectangle
2. Draw the invariant visual components of the symbol

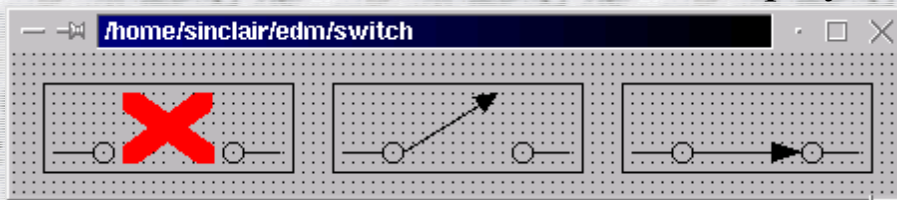


3. Copy this information and paste it N times, you now have N+1 visual states



Creating Symbols (cont)

4. Draw the state dependent visual components, the first state should be the out-of-band state, the second state is displayed in edit mode



5. Make sure no grouped objects exist, click the middle mouse button on the display background, and choose *Auto make symbol* from the menu
6. Save the EDM display file, this file may now be used as a symbol file

Deploying Symbols

- A symbol instance is created like any other EDM object
- One property of a symbol instance is the symbol file name; this is the file discussed previously
- An exercise will illustrate this entire process in detail

Exercise 4 (1 of 3)

■ Objective:

- Create symbol template file
- Explore symbol dynamic behavior

■ Execute edm and create a new display

■ Draw 3 symbol states as follows

- Create invisible rectangle
- Draw invariant symbol components
- Copy image and paste two copies to the display
- Draw state dependent components

Exercise 4 (2 of 3)

- Arrange images in a rows/columns ordering, first state is upper-left, last is lower-right
- If any objects have been grouped, ungroup now
- Click middle button, choose *Auto make symbol*
- Save symbol file as symbol1

■ Open example1

■ Add a symbol instance

- Use symbol file recently created, use same PV as referenced in text control object
- State 1: $1 \leq \text{PV value} < 0$
- State 2: $2 \leq \text{PV value} < 1$

Exercise 4 (3 of 3)

- Save the display file
- Execute the display, change value of the PV, and observe the symbol behavior
- Experiment with symbol color options

Graphics

- Rectangle
- Circle
- Arc
- Lines
- Text
- Image - GIF & PNG
- Dynamic Symbol

Monitors

- Analog Meter
- Bar Meter
- Byte
- Strip Chart
- Text
- Symbols

Controls

- Text
- Slider
- Buttons
 - binary, message, menu, up-down, exit
- Message Box
- Related Display
- Shell Command

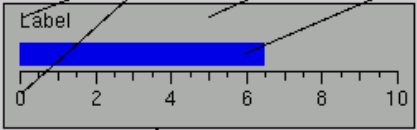
Exercise 5

- Obtain several PV names from instructor
- Experiment with various graphic, monitor, and control objects

Questions

- How do I ...

Bar

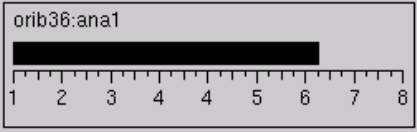


Fg Color Bg Color Bar Color

Label

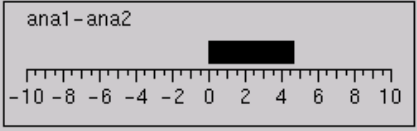
Border: Yes
Show Scale: Yes
Label Type: Literal
Origin: 0

Border



orib36:ana1


Border: Yes
Show Scale: Yes
Label Type: PV Name
Origin: 1



ana1-ana2

Border: Yes
Show Scale: Yes
Label Type: Literal
Origin: 0
Readback & Null PV specified

Bar shows Readback - Null



Border: No
Show Scale: No
Label Type: Literal
No Label string specified
Origin: 0

Scale Format Options

- GFloat - precision gives number of significant digits
- FFloat - precision gives number of decimal places
- Exponential - precision gives number of decimal places, number is displayed in scientific notation

If Scale Info From DB is checked, then scale precision, min, and max are taken from HOPR & LOPR fields in database.

Static Text

Static Text

Text String Fg Color
Bg Color
Use Display Bg: Yes

Text String Fg Color
Bg Color
Use Display Bg: No

Text String Auto Size: No, left justified

Text String Auto Size: No, right justified

Text String Auto Size: No, center justified

Note: The dashed lines around the text strings have been added to show the actual object size boundaries.

Text string may contain symbols.

Auto Size sets the geometrical boundary of the text string and should not be used with center and right justification.

Color PV is used with dynamic colors and alarm sensitivity. If both are present, alarm colors have precedence.

Visibility may be achieved through visibility PV and embedded rule or with invisible color.