MEDM Reference Manual

Kenneth Evans, Jr.

September 1999

Advanced Photon Source

Argonne National Laboratory

9700 South Cass Avenue, Argonne, Illinois 60439

This document is always under construction in order to incorporate new features in MEDM as they are added. Sometimes these features may not yet be available in the currently-released version of MEDM.

Contents

Overview

MEDM stands for Motif Editor and Display Manager. It is a graphical user interface (GUI) for designing and implementing control screens, called displays, that consist of a collection of graphical objects that display and/or change the values of EPICS process variables. The supported objects include buttons, meters, sliders, text displays/entries, and graphs. It has two modes of operation, EDIT and EXECUTE. Displays are created and edited in EDIT mode, and they are run in EXECUTE mode. MEDM is an EPICS extension.

New Features

Version 2.3

2.3.6

- The Image has been enhanced to read GIF89a files (the latest GIF specification). It now handles multiple-frame GIFs. Multiple-frame GIFs can be set to animate or to display a specific frame based on a CALC expression that can include up to four process variables. This opens a wide range of creative possibilities, including implementing Color Rules. There is also a new EPICS extension called Simple GIF Animator (SGA) that allows you to create and change animated GIFs.
- The Dynamic Attribute used by graphics objects has been extended to have its visibility be based on a CALC expression that can include up to four process variables and more.
- The Composite now has a Dynamic Attribute. This means whole sections of the display can be made to appear, disappear, or change based on the values of process variables. Also, by grouping any single object, it can be made to be Composite and therefore have a Dynamic Attribute. The objects for a Composite may now be read from an ADL file in addition to being defined through grouping.
- There is a siteSpecific.h file, where certain defaults may be defined when MEDM is built.
- Colormaps for a display may be specified in the ADL file, a separate file from the ADL file, or by using the default. The separate-file capability was present before, but was broken. The default display colormap may be specified in siteSpecific.h. Different displays may, as before, have different colormaps.

- Unix type builds are deprecated, which means they may still work but should be avoided.

2.3.5

- The limits and the precision for objects such as Meters, Sliders, and Text Updates can be specified at design time and changed at run time, instead of being set by fields such as HOPR, LOPR, and PREC in the associated record.
- The Menu and Choice Button have been fixed so their size is what is specified at design time and what is stored in the ADL file. They are now the same size in both EDIT and EXECUTE mode and have the same look as other MEDM objects. The objects are now smaller than they used to be. The impact is more noticeable for small versions of these objects, and they may need to be resized on existing screens to look attractive.
- There is an alternative, SciPlot, to the commercial package, XRT/Graph, used for the Cartesian Plot. SciPlot does not have all of the features of XRT/Graph, but it does have enough features to use for most plotting needs. It has the advantage of being free and available for platforms where XRT/Graph does not exist. The WIN32 version of MEDM now supports the Cartesian Plot.
- The DESC and RTYP fields of a process variable have been added to the PV Info Dialog Box.

- There is a "Save All" option on the File menu.
- There is a command-line option (-noMsg) to prevent the Message Window from popping up.
- The shadows on all items with shadows are now consistent. Highlights have been removed from all objects. Sizes are consistent in EDIT and EXECUTE mode and are what is specified in the ADL file except for minimum size requirements.
- MEDM is supported for CDEV.
- MEDM can be built with either Host or Unix type builds.

2.3.4

- The default startup option is -local. There is a new option, -attach, to either attach to an existing MEDM if there is an eligible one, or to make the current one eligible for attachment if there is not. You need to use -attach to achieve the former default startup behavior. This behavior is primarily applicable for a control room, where it is typically set in scripts. It was confusing and not appropriate for individual users.
- The overlapping rectangles on the Related Display label do not appear if the label text starts with "-".
- The Meter, Bar Monitor, and Scale Monitor have been fixed to be the size specified. Formerly, they were two pixels wider, this space being used for the border. There is now no border, and the top and bottom shadows are the same as for Motif widgets, rather than being drawn with another, simplified algorithm. The Bar Monitor appears raised. It is not expected these changes for consistency will significantly affect most display screens.

- The Bar Monitor has a no decorations option. This allows making bar graphs.
- MEDM is available in a beta version for the WIN32 platform.

2.3.3

- Help is now functional. Help from the menus and context sensitive help uses Netscape. If Netscape is not up, it takes a little while for it to come up after the first help request; however, once it is up, new help topics appear quickly inside the existing Netscape. Bubble help has been added to the Object Palette to identify the objects.
- There is a greatly expanded MEDM reference manual in both hardcopy and HTML versions. You are probably looking at one or the other of them now.
- There is now an Undo.
- New editing operations through both the mouse and the keyboard have been added. You can now select objects that are under other objects and can resize with Ctrl-Arrow keys.
- An optional grid has been added for editing. It can be toggled on and off for each display, and the spacing is adjustable. Snap to grid has been implemented.
- "Undo", "Align", "Space Evenly", "Center", "Orient", "Size", "Grid", "Find Outliers", "Refresh", and "Edit Summary" have been added to the Edit menu.
- You can now change the foreground and background colors simultaneously for all grouped elements.
- There is a new PV Info feature.
- There is a new Execute Menu item on the Execute-Mode Popup Menu.
- A number of modifications have been made to the Related Display including executing it without showing the menu if there is only one item, the ability to replace the parent display, the ability to have labels rather than the overlapping squares on the menu button, and the ability to have a hidden button.
- The Cartesian Plot has a time format for the x axis, and there are several new user interactions, such as zoom, in EXECUTE mode.
- Shell commands, as well as commands on the Execute Menu, now accept special characters to include a process variable, include the name of the ADL file, or to prompt for user input.
- The MEDM main window, object palette, resource palette, and color palette now have fixed, default locations. You can change these locations with resources. See Initial Window Locations.
- The way in which MEDM starts up and connects to an existing MEDM has been made more robust. There should be minimal need to use -cleanup.
- You no longer have to respond to the "Do you really want to Exit?" dialog box. MEDM will still prompt you to save any unsaved files, however.
- Much of the coding has been rewritten to be clearer and better organized, and many bugs have been fixed.

History

MEDM was started by Mark Anderson at Argonne in 1990. It was originally partially based on the two programs, DM (Display Manager) and EDD (Edit Display), which had been developed at Los Alamos. These programs used the X library directly. MEDM handles both the construction and running of displays and has a Motif interface. Los Alamos has continued to develop DM and EDD, and today they have similar features to MEDM, but a different look and feel. The basic design and most of the development MEDM was done by Mark. Mark was a strong supporter of Motif in the days before it was as much of a standard as it is today.

Fred Vong took over MEDM in the fall of 1994, starting with Version 1.4.6. He made the program more responsive under heavy load and added a new ADL file format that used less space. Among other things, he added Channel Access security, rewrote the Strip Chart, added features to the Cartesian Plot, and added to the functionality of the Related Display. Fred was rewriting a large part of the coding to make it cleaner and improve the logic when he left Argonne in late 1996.

Ken Evans took over when Fred left starting with Version 2.3.0. He merged Fred?s development version with the distributed version, and finished debugging what Fred had started. He wrote this reference manual and is responsible for the new features described above, as well as for substantial reorganization and cleanup of the code.

In addition to these primary developers, parts of MEDM have come from many members of the EPICS community and from many people in the computing community who provide public-domain versions of their routines. Mark Rivers of the University of Chicago did the original port to the WIN32 platform and has provided much help in getting MEDM running on several other platforms. Jie Chen of the Thomas Jefferson National Accelerator Facility provided the initial implementation of SciPlot and all of the CDEV support.

Requirements

MEDM requires X Windows, Motif and EPICS. Motif Version 1.2 or above is supported. MEDM attempts to conform to the Motif Style Guide. Hence standard mnemonics and accelerators are available for interface navigation. Also help is available in standard forms, including context-sensitive help. Netscape is required for most of the MEDM help.

MEDM is developed and tested in an environment using the Solaris operating system and the TED desktop environment as distributed by TriTeal Corporation. TED is a desktop environment conforming to the Common Desktop Environment (CED) standards and uses Motif Version 3. At this time MEDM has no special enhancements to support CDE features apart from Motif or features that are special to Motif Version 3. The main parts of the program build with no compiler warnings even with CMPILR=STRICT on this system. The program should run with versions of Motif 1.2 or higher, but it may need modifications to run on other operating systems. We will try to incorporate changes necessary to support other systems if you let us know.

Command Line

MEDM can be executed from the command line in the general form:

medm [X options]

```
[-help | -?]
[-version]
[-x | -e]
[-local | -cleanup | -attach]
[-cmap]
[-bigMousePointer]
[-noMsg]
[-displayFont font-spec]
[-macro "xxx=aaa,yyy=bbb, ..."]
[-dg [xpos[xypos]][+xoffset[+yoffset]]
[display-files]
[&]
```

Parameters in square brackets [] are optional. Italicized parameters are to be replaced by user-specified terms. The other parameters should be typed literally. The command-line options are described below. The "&" causes the process to be run in the background, usually a good idea for a graphical program.

Command-Line Options

X options	The usual X command-line arguments, such as -display and -geometry.
-help or -?	Print usage.
-version	Print the version.
-x	Startup MEDM in EXECUTE mode.
-e	Startup MEDM in EDIT mode. (This is the default.)
-local	Do not participate in remote display protocol: Do not transfer operation to an already-runnin MEDM, and do not allow transfer of operation from any other MEDM. (This is the default.)
-attach	Do participate in remote display protocol: Transfer operation to an existing MEDM if there i one which is accepting transfers, or have this one assume responsibility for accepting transfers if there is not one.) This can lead to more efficient operation and is appropriate if many MEDM screens are running on one display, such as in a control room. You must give the first MEDM time to establish itself before attempting to attach to it. See Smart Startup.
-cleanup	Participate in remote display protocol, but ignore any existing MEDM and take over responsibility for accepting remote transfers.
-cmap	Use a private X colormap to circumvent the problem of not having enough colors available i the colormap. (This option may cause colormap flashing.)
-bigMousePointer	Use big mouse cursor.
-noMsg	Set the Message Window to not pop up when messages are received. (It can be brought up via the View menu on the MEDM main window if you need to see it.)
-displayFont font-spec	Select aliased or scalable fonts. (See Fonts below.) The font-spec parameter can be "alias", "scalable", or an X font specification. When attaching, this argument should be the same for all participating MEDMs.
-macro name=value	Apply macro substitution to replace occurrences of $(name) with value in all command-line specified display list files.
-dg geometry	Specify the geometry (location and size) of the display window. The geometry syntax is similar to the X Windows -geometry option. (Note that -geometry specifies the location of the MEDM main window, and -dg specifies the location of the display windows.) The -dg option affects the ADL filenames following it.
display-files	Specify list of display (ADL) files to use in EXECUTE mode. (This option requires -x also.)

Examples

Start up in default EDIT mode:

```
medm &
```

Start up executing two displays:

```
medm -x abc.adl def.adl &
```

Start up in EDIT mode with a new, local executable:

```
medm -local &
```

Start up using default scalable fonts:

```
medm -displayFont scalable &
```

Start up in execute mode, performing macro substitution of all occurrences of $(a) and $(c) in the display file, xyz.adl:

```
medm -x -macro "a=b,c=d" xyz.adl &
```

Start up in execute mode; resize the display window corresponding to abc.adl to 100 by 100 pixels and move it to the location, x = 100 and y = 100; and move the display window corresponding to def.adl to x = 400 and y = 150:

```
medm -x -dg 100x100+100+100 abc.adl -dg +400+150def.adl &
```

ADL Files

The files that contain the information for each display are called ASCII Display List (ADL) files and should have an extension of .adl. These files are created or modified when you save a display that you have edited. The files are readable and can also be edited by hand if you understand the file format. The file format is not described here, but it should not be difficult to make many types of changes without knowing it precisely. MEDM should be able to read files that were written in earlier formats, and files can be saved with the file format in use for Versions 2.1.x as well as the default format, which was initiated with Version 2.2.0.

The following logic is used to open ADL files from the command line and from Related-Display execution:

1. Try to open the file with the name as given. If the name is not an absolute pathname, then this means to try to open the file relative to the current working directory (the directory where MEDM was started).
2. If the file is being opened as a result of a related-display button, then try to open the file with the name prepended with the pathname of the display that contained the button.
3. Try to open the file with the name prepended by each pathname in the EPICS_DISPLAY_LIST.

Site-Specific Customization

Certain defaults, such as fonts and colors, may be specified in the siteSpecific.h file when MEDM is built. This provides an alternative to using the defaults in use at the Advanced Photon Source. Some of these defaults were decided a long time ago and cannot be changed because it would break too many existing screens. Often there are better alternatives today. In particular, the use of scalable fonts is highly recommended. See the siteSpecific.h file in the MEDM distribution for more information.

Fonts

MEDM uses several methods for font specification, invoked on the command line by the option:

```
-displayFont font-spec
```

where *font-spec* is one of {"alias", "scalable", or an X font specification, such as -bitstream-courier-medium-r-normal--0-0-0-0-m-0-iso8859-1}. The default is -displayFont alias.

Aliased Fixed Fonts

For default, fixed fonts, MEDM uses font aliases internally. This makes it possible to deal with server font variations. The aliased fonts are not in the default X database. To make these fonts available you need to do the following:

Aliased Fonts in X Windows

Copy the following lines into your miscellaneous fonts.alias file. The standard place is /usr/lib/X11/fonts/misc/fonts.alias. You can get your own particular font path from "xset q". Note that these specifications are the standard MEDM font specifications as used at the Advanced Photon Source, but you are free to define another set of font specifications that gives better results for your machine or your own preferences. Keep in mind that displays may not look good if the fonts used are significantly different from those used by the designer of the display.

```
widgetDM_4 -misc-fixed-medium-r-normal--8-60-100-100-c-50-iso8859-1

widgetDM_6 -misc-fixed-medium-r-normal--8-60-100-100-c-50-iso8859-1

widgetDM_8 -misc-fixed-medium-r-normal--9-80-100-100-c-60-iso8859-1

widgetDM_10 -misc-fixed-medium-r-normal--10-100-75-75-c-60-iso8859-1

widgetDM_12 -misc-fixed-medium-r-normal--13-100-100-100-c-70-iso8859-1

widgetDM_14 -misc-fixed-medium-r-normal--14-110-100-100-c-70-iso8859-1

widgetDM_16 -misc-fixed-medium-r-normal--15-120-100-100-c-90-iso8859-1

widgetDM_18 -sony-fixed-medium-r-normal--16-120-100-100-c-80-iso8859-1

widgetDM_20 -misc-fixed-medium-r-normal--20-140-100-100-c-100-iso8859-1

widgetDM_22 -sony-fixed-medium-r-normal--24-170-100-100-c-120-iso8859-1

widgetDM_24 -sony-fixed-medium-r-normal--24-170-100-100-c-120-iso8859-1

widgetDM_30 -adobe-times-medium-r-normal--25-180-100-100-p-125-iso8859-1

widgetDM_36 -adobe-helvetica-medium-r-normal--34-240-100-100-p-176-iso8859-1

widgetDM_40 -adobe-helvetica-bold-r-normal--34-240-100-100-p-182-iso8859-1

widgetDM_48 -adobe-helvetica-bold-r-normal--34-240-100-100-p-182-iso8859-1

widgetDM_60 -adobe-helvetica-bold-r-normal--34-240-100-100-p-182-iso8859-1
```

Then run "xset fp rehash" to install them. You can see if the fonts are there with "xlsfonts" or "xlsfonts | grep widgetDM". You can get more information from any of the general-purpose books on X Windows.

Aliased Fonts in Exceed (An X Server for Microsoft Windows)

1. Copy the same lines to c:\Exceed\user\misc.ali. (Assuming Exceed is installed in c:\Exceed.) I t is probably necessary to have 100 dpi fonts installed, which may not be the default (depends on screen size). Otherwise the larger font sizes may not be available.
2. In the Xconfig program pick **Fonts**, then **Font Database...**. Then push **Rebuild Database**. (In Windows 95 you get Xconfig from the **Start/Programs/Exceed** menu or by right clicking on **Exceed** when it is on the task bar and choosing **Tools/Configuration**.)
3. You may be able to right click on Exceed if it is on the task bar and choose **File, Reload Database, Font**. Otherwise restart Exceed.

Alternatively, you can add a font server that is known to have the fonts:

1. In the Xconfig program pick **Fonts**, then **Font Database...**. Then push **Add;** select **Server**; and configure the server by filling in the dialog box. See your system manager for the parameters, which may be something like tcp/phoebus:7100.
2. You may be able to right click on Exceed if it is on the task bar and choose **File, Reload Database, Font**. Otherwise restart Exceed.

Aliased Fonts in XONNET (An X Server for Microsoft Windows)

Copy the lines to XONNET32\FONTS\MISC\FONTS.ALI. Further information on fonts is described in the program's help.

Default Scalable Fonts

The user can invoke MEDM with the -displayFont scalable option. MEDM then uses the default Speedo outline font (bitstream) supplied by the X11R5 font server. Users should add a font server to their font path via:

```
% xset +fp tcp/:
```

for example:

```
% xset +fp tcp/phebos:7100
% medm -displayFont scalable
```

You can get the name of the font server by running "fsinfo" on the machine that has the server.

User-Specified Scalable Fonts

The user can invoke MEDM with the -displayFont option. MEDM then uses the specified font supplied by the X11R5 fontserver. This font should be an XLFD name (all 14 hyphens, other fields can be wildcarded though) and scalable (i.e., point and pixel size = 0). Users should add a font server to their font path as above.

For example:

```
% medm -displayFont -bitstream-courier-medium-r-normal--0-0-0-0-m-0-iso8859-1
```

Initial MEDM Window Locations

The MEDM main window, the object palette, the resource palette, and the color palette are located at the right edge of the screen by default. The spacing between them depends on the window manager title bar and may not be correct for all window managers. They can be changed using the standard format for X Windows geometry by the following resources (shown with typical values):

```
Medm.mainMW.geometry: -5+5
Medm.objectS.geometry: -5+137
Medm.resourceS.geometry: -5+304
Medm.colorS.geometry: -5-5
```

You could put these lines in your .Xdefaults file, for example.

Palettes

Object Palette

Resource Palette

Color Palette

MEDM Objects

The objects that can be used in MEDM fall into four categories, Graphics, Monitors, Controllers, and Miscellaneous.

Graphics are items such as text, lines, and images. Monitors are objects that monitor the state or values of process variables. Controllers are objects that change the values of process variables. There is only one miscellaneous object, the Select Arrow, which is not really an object but an editing mode.

Some objects are widgets. This means they have their own X window, which is a child of the drawing area of the display. The Graphics objects, such as the Rectangle, Oval, Image, *etc.* are drawn on the drawing area. Every object with a widget lies above the drawing area and hence above all drawn objects. Further, all graphics objects that are updating because their visibility depends on a process variable or because they are an animated Image will appear above non-updating graphics objects. If you want a non-updating object to be properly stacked among updating objects, associate it with a process variable but leave its Visibility Mode as Static.

In EDIT mode, all of these objects are represented on the Object Palette, and this palette can be used to create new instances of the objects on the display.

Object Index

Graphics	**Monitors**	**Controllers**	**Special**
Arc	Bar Monitor	Choice Button	Composite
Image	Byte Monitor	Menu	Display
Line	Cartesian Plot	Message Button	
Oval	Meter	Related Display	
Polygon	Scale Monitor	Shell Command	
Polyline	Strip Chart	Slider	
Rectangle	Text Monitor	Text Entry	
Text			

Attribute Index

Attributes are sets of properties. These sets of properties may be common to several, or even most, objects. Simple attributes have one property that can take on one of several values. Compound attributes have more than one property. To prevent repetition, the sets of attributes are described here, rather than with the objects that possesses them. The compound attributes are listed first, then the simple objects in alphabetical order.

Compound	Simple		
Object	Cartesian Plot Axis Style	Edge Style	Related Display Mode
Basic	Cartesian Plot Range Style	Erase Mode	Related Display Visual
Dynamic	Cartesian Plot Style	Fill Mode	Stacking
Monitor	Cartesian Plot Time Format	Fill Style	Text Align
Control	Color Mode	Image Type	Text Format
Limits	Color Rule	Image Calc	Time Units
Plot	Direction	Label	Visibility Mode
Plot Axis		Plot Mode	Visibility Calc

Objects

Arc

The Arc has Object (X Position, Y Position, Width, Height), Basic Attribute (Foreground, Style, Fill, Line Width), and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. In addition, there is a Begin Angle and a Path Angle. The Arc is drawn on the display drawing area and is not a widget. The Begin Angle is the start of the arc in degrees. The Path Angle is the length of the arc in degrees. (Internally the angles are specified in integer 1/64-degree units.)

Bar Monitor

The Bar Monitor has Object (X Position, Y Position, Width, Height), Monitor (Readback Channel, Foreground, Background), Limits (Low Limit, High Limit, Precision), Label, Direction, Fill Mode, and Color Mode attributes.

Byte Monitor

The Byte Monitor has Object (X Position, Y Position, Width, Height), Monitor (Readback Channel, Foreground, Background), Direction, and Color Mode attributes. It also has a Start Bit and End Bit.

The Byte Monitor is designed to monitor mbbiDirect and mbboDirect records. It was developed at the Thomas Jefferson National Accelerator Facility (formerly CEBAF) by David M. Wetherholt.

The Start Bit and End Bit are integers in the range 0-15, specifying the starting and ending bits to be displayed. The End Bit can be smaller than the Start Bit.

Cartesian Plot

The Cartesian Plot has Object (X Position, Y Position, Width, Height), Plot (Title, X Label, Y Label, Foreground, Background), Plot Style, Plot Mode, and Erase Mode attributes. In addition, there is a Count, X/Y/Trace Data, Axis Data, a Trigger Channel, and an Erase Channel, which are described in this section.

The logic of the Cartesian Plot is separate from the plotting package that implements it. At present MEDM supports two implementations.

XRT/Graph

The original and primary implementation is by the commercial plot package, XRT/graph by the KL Group. MEDM supports both Versions 2.4 and 3.0 of XRT/Graph. The implementation of these two versions is significantly different, but the plots look the same. The XRT/graph package must be purchased. To build MEDM with XRT/Graph, the makefile variable XRTGRAPH must be set to the path for XRT/Graph. After the usual compile and link, the executable must be patched to enable the Cartesian Plot. To do this, the proper, licensed XRT patch files must be available. If these files exist on the computer on which MEDM is built, the MEDM Makefile will do the patch as part of the build. The patch can also be done later on a computer with the proper files. XRT/Graph is a very flexible and full-featured plotting package. One can assume it has been tuned for performance.

The following user interactions are available in EXECUTE mode for the Cartesian Plot when built with XRT/graph 3.0 and when XRT_EXTENSIONS is defined in the Makefile:

Ctrl-Btn1	Graphics zoom. Drag to select the area of the graph and release to zoom. This action blows up the graph to only show the area in the rubberband rectangle.
Shift-Btn1	Axis zoom. Drag to select the range of the axes and release to zoom. This action changes the ranges of the axes and repositions the axes so they are visible.
Ctrl-Btn2	Scaling. Move up or down to increase or decrease the size.
Shift-Btn2	Translation. Move to shift the graph.
r	Press r to reset the graph from changes made with Btn1 and Btn2.
Btn3	Bring up the Cartesian Plot Axis Data dialog box to change axis parameters.
Ctrl-Btn3	Bring up the XRT Property Editor dialog box. The property editor allows you to change most of the graph properties. The editor is complex, but the operation is intuitive. Select the upper tab that corresponds to the element of the graph you want to edit. If there is a tree control on the left, select the specific object to edit, then find and edit the property on the lower tabs. Keep in mind that it is possible to confuse MEDM by changing the properties this way, since the changes are internal and not necessarily known to MEDM. Consequently, the use of this feature is not supported. It can be useful if used sensibly, however. The graph can be restored by closing and reopening the display. For more information about the property editor and the various graph properties, see *the XRT/graph Programmer's Guide and Reference Manual, Version 3.0*.

SciPlot

MEDM also supports SciPlot, a freely distributable package written by Rob McMullen at the University of Texas. Its home page is http://www.ae.utexas.edu/~rwmcm/SciPlot.htmlThe version used in MEDM has been modified to add features and fix bugs, but is essentially the work of Rob McMullen. There is a license, which is included in the files and discussed on the web site. At present, SciPlot does not support a second Y axis, fill, or time labels, as well as some less-important features. However, it has enough features to be very useful, and it does some things better than XRT/Graph. It can be used where XRT/Graph is not available, for instance on the WIN32 platform. To build MEDM with SciPlot, define the makefile variable SCIPLOT, preferably setting it to YES.

If neither of the makefile variables XRTGRAPH or SCIPLOT is defined, MEDM can still be built but will ignore any Cartesian Plots. Any Cartesian Plots in an existing ADL file will be lost if the file is modified and saved.

The Cartesian Plot is the most complicated MEDM object. It has many options and can be confusing. The descriptions presented here are intended as a reference, not as a tutorial. A good approach is to change the various options and see what happens. It is usually a good design choice to keep the plot as simple as possible.

An MEDM Cartesian Plot consists of an X and one or two Y axes on which data can be plotted. The sets of data are called traces and consist of a set of (x, y) pairs and a color. The traces correspond to curves on the plot. Currently there can be up to eight traces on a plot. The first trace is plotted on the Y1 axis, and the remaining traces (if any) are all plotted on the Y2 axis. The X and Y1 axes may have a label, but the Y2 axis does not. Each trace can (but does not have to) have a process variable from which to get the x values and another from which to get the y values. These process variables can be array process variables, such as Waveforms, or they can be scalar process variables with only a single value.

There are eight possible kinds of traces as seen in the following table. The traces for a given plot do not all have to be of the same kind - they can be mixed. (In the table Nx is the number of elements in the process variable specified for x, and Ny is the number of elements in the process variable specified for y. The letter n denotes a number greater than one, and a blank indicates no process variable is specified. The axis limits LOPR and HOPR denote the limits obtained from Channel Access for the process variable. Typically, these are the fields LOPR and HOPR in the associated record. Count is the specified Count for the Cartesian Plot, which is described in more detail below.)

Kinds of XY Traces

Nx	Ny	Type	Xmin	Xmax	Ymin	Ymax	N Po
n	n	X,Y Vector	LOPR	HOPR	LOPR	HOPR	Min(Nx, N
n	1	X Vector, Y Scalar	LOPR	HOPR	LOPR	HOPR	Nx
1	n	Y Vector, X Scalar	LOPR	HOPR	LOPR	HOPR	Ny
1	1	X,Y Scalar	LOPR	HOPR	LOPR	HOPR	Count
n		X Vector	LOPR	HOPR	0	Count - 1	Nx
1		X Scalar	LOPR	HOPR	0	Count - 1	Count
	n	Y Vector	0	Count - 1	LOPR	HOPR	Ny
	1	Y Scalar	0	Count - 1	LOPR	HOPR	Count

If one of the process variables is an array with more than one element, then the specified Count is ignored. Otherwise, Count corresponds to a history of the process variable. Each time the process variable changes (either one, in the case of X, Y Scalar) a new point is plotted until there are Count points. When the Plot Mode is "plot n pts & stop," no more points are plotted. When the Plot Mode is "plot last n pts," the earliest point is discarded, the others are moved down, and the latest is plotted at the end. In the cases where one of the process variables is not specified, the history is plotted on that axis with values from 0 to Count -1.

The X/Y/Trace Data are entered via the Cartesian Plot Data dialog box with entries for the process variables associated with x and y and an entry for the color. The color is picked by clicking on the color entry and using the color palette. The process variable names can both be left blank on a given row, in which case the row is ignored. The first trace (the one that uses the Y1 axis) is the first valid one, and does not have to be in the first row. The data entry is different from that for entry boxes in the resource palette. Typing is not lost, for example, when the entry box loses focus, and it is not necessary (or useful) to type a carriage return to retain the data. The data are retained only when the apply button is pressed.

The Axis Data are entered via the Cartesian Plot Axis Data dialog box. This same dialog box is available in EXECUTE mode by clicking Btn 3 on the Cartesian Plot. The dialog box consists of an area for each of the three axes (X, Y1, and Y2). For each axis there is a Plot Axis (Axis Style, Axis Range, Minimum Value, Maximum Value, Time Format) attribute. The maximum and minimum values are only relevant if the Axis Range is "user-specified", and the time format is only relevant for the X axis when the Axis Style is "time". The values for the maximum and minimum values are only retained if a carriage return is typed in the box, although typing is not lost when the entry box loses focus.

In order to use the time format meaningfully, the process variable data must be in suitable time units. The basic time unit is a second. The EPICS time base, corresponding to zero seconds, is 00:00:00 January 1, 1990, Coordinated Universal Time (UTC), formerly known as Greenwich Mean Time (GMT). This is the time base used in EPICS timestamps, and it is *assumed* to be the time base for any process variable data. Note that it is not the same as local time. The time base in MEDM is set to 00:00:00 January 1, 1990, local time. By setting the time base as local time in MEDM, the effect is to make the EPICS UTC time numbers appear as local time, rather than the less intuitive UTC. For times less than one day and for time formats that do not include the day or year, the data could just as well be seconds since midnight UTC, since the plotted points on this type of axis would be indistinguishable from those plotted with the actual base.

It should be noted that there is an error introduced in converting the number of seconds into a floating point number for plotting. This error corresponds to about 2 minutes per year, amounting to 20 min. by the year 2000.

The Trigger Channel is a process variable that causes the entire plot to be updated. If there is a trigger channel, the plot is updated whenever the value of that process variable changes. Otherwise, each individual trace is updated whenever any of the data for that trace changes.

The Erase Channel is a process variable that causes erasing of the plot. If there is an Erase Channel, the plot erases when the process variable turns either zero or non-zero, depending on the Erase Mode. The Erase Mode is only relevant if there is an erase channel.

Choice Button

The Choice Button has Object (X Position, Y Position, Width, Height), Control (Control Channel, Foreground, Background), Color Mode, and Stacking attributes. The Choice Button is used for ENUM process variables and is a means for picking the ENUM choices via buttons.

Display

The Display is a special object. It has Object (X Position, Y Position, Width, Height) and Grid attributes. In addition there are Foreground and Background colors, and a Colormap, which is the name of a display colormap file. See Grid for a description of the parameters in the Grid attribute, and see Display Colors for a description of display colormaps and colormap files. The Background color is the color of the display background, and the Foreground color is the color of the grid.

The X and Y positions are saved in the ADL file. The display should appear at the coordinates specified in the ADL file when the display is first brought up. They are updated in the Display object only when the Display is selected in the resource palette by clicking on the display background in EDIT mode. They are not updated when you drag the display to new positions. If you want to have the display come up at a specified position, be sure the X and Y positions are what you want when the file is saved.

Image

The Image has Object (X Position, Y Position, Width, Height) and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. Color Mode in the Dynamic Attribute is not relevant and is not used. There are also Image Type, Image Name, and Image Calc attributes. The Image is drawn on the display drawing area and is not a widget.

MEDM only supports Graphics Interchange Format (GIF) image files, both the older GIF87a and the newer GIF89a formats. The GIF image files may include multiple frames, transparency, and most other features supported by the GIF format except local color maps. Transparency only substitutes the display background color for the transparent color. Objects stacked under the image do not show through. The Image Name attribute is the name of the GIF file. When an Image is inserted in EDIT mode, a file selection box prompts the user for the image file to be incorporated.

If the GIF image file has multiple frames, the designer can choose to animate the images or to display a specified frame. The default is to animate if there is more than one frame. To display a specific frame, enter an expression for the Image Calc that returns a frame number. The syntax for the Image Calc expression is the same as that for the Dynamic Attribute, and it uses channels A-D in the Dynamic Attribute. The only difference is that the CALC expression should return a frame number, not True or False. The value will be rounded to the nearest integer. Frame numbers that are too high will use the last frame, and frame numbers that are too small will use the first frame. The Image will be black in EXECUTE mode if the Image Calc expression is invalid.

The visibility of the Image is determined in the usual way by the Dynamic Attribute if there is a channel specified. The visibility is independent of whether the image is animating or not.

A single-frame Image with no process variables specified for the Dynamic Attribute is treated the same as other Graphics Objects. It is drawn on the display background and will be under any objects that are updating. A multiple-frame Image even with no process variables specified for the Dynamic Attribute does update and so will appear among the updating objects in the stacking order and hence over non-updating objects. This is a concern only for overlapping objects. See the introduction to MEDM Objects for more information on stacking order.

A GIF image may contain up to 256 colors. On an 8-bit X display there are only 256 available colors in the colormap, some of which are almost certainly being used. Consequently, MEDM has an algorithm to reduce the number of colors in the image if the X colormap is full. This algorithm makes the image appear less bright. Images may also appear to have different brightness at different times, depending on the state of the X colormap when the MEDM display is loaded. There should be no problem for a 24-bit X display or when MEDM has its own colormap. See the command-line options for how to specify a private colormap for MEDM.

Color Rules may be implemented using the Image. To do this, place your object over an Image that is large enough to provide a border around the object. You can use a GIF image file that has one frame for each color you want to display. These frames only need to be one pixel wide and high in the GIF file, saving storage space and loading time. MEDM will resize them as it does any image. Use the Image Calc expression and the process variables in the Dynamic Attribute to specify when to display each color.

There is an EPICS extension called Simple GIF Animator (SGA) that allows you to create and modify animated GIFs for use in MEDM. There are also many other GIF animators available, especially for WIN32. It does not make any difference which platform you use to create or manipulate the images. You cannot use SGA or most of the animators to create the images themselves, only to add, delete, and rearrange them plus modify some of the GIF parameters. There are many drawing and image editing programs that allow you to create and edit images. For UNIX, the GNU Image Manipulation Program (GIMP) is a good choice and is free.

Line

A Line is a Polyline with just two points. See Polyline.

Meter

The Meter has Object (X Position, Y Position, Width, Height), Monitor (Readback Channel, Foreground, Background), Limits (Low Limit, High Limit, Precision), Label, and Color Mode attributes. It displays the value of the process variable on a meter with a dial.

Menu

The Menu has Object (X Position, Y Position, Width, Height), Control (Control Channel, Foreground, Background), and Color Mode attributes. The Menu is used for ENUM process variables and is a means for picking the ENUM choices via a menu.

Message Button

The Message Button has Object (X Position, Y Position, Width, Height), Control (Control Channel, Foreground, Background), and Color Mode, attributes. In addition, there are a Message Label, a Press Message, and a Release Message.

The Message Label is the label on the Message Button. The Press Message and the Release Message are the values to which to set the process variable when the button is pressed or released, respectively. These values should be commensurate with the type of the process variable. (It is not wise to send a non-numeric string to a DOUBLE process variable, for example.) The label and message strings can be up to 255 characters.

Oval

The Oval has Object (X Position, Y Position, Width, Height), Basic Attribute (Foreground, Style, Fill, Line Width), and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. The Oval is drawn on the display drawing area and is not a widget.

Polygon

The Polygon has Object (X Position, Y Position, Width, Height), Basic Attribute (Foreground, Style, Fill, Line Width), and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. The Polygon is drawn on the display drawing area and is not a widget.

Polygons are drawn by pressing Btn1 to generate vertices or Shift-Btn1 to generate vertices which are multiples of 45 degrees from the previous point. This allows easy horizontal or vertical line generation, for instance. Button releases are ignored. The Polygon is finished when there is a second click at the same location. The Polygon is always displayed as closed while it is being drawn.

Polygons can be edited by selecting them, then pressing Btn1 on a vertex and dragging the vertex.

Note that Polygons in outline fill mode are similar to Polylines, except that Polygons close the figure and Polylines do not.

Polyline

The Polyline has Object (X Position, Y Position, Width, Height), Basic Attribute (Foreground, Style, Line Width), and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. The Polyline is drawn on the display drawing area and is not a widget.

Polylines are drawn by pressing Btn1 to generate vertices or Shift-Btn1 to generate vertices which are multiples of 45 degrees from the previous point. This allows easy horizontal or vertical line generation, for instance. Button releases are ignored. The Polyline is finished when there is a second click at the same location.

Polylines can be edited by selecting them, then pressing Btn1 on a vertex and dragging the vertex.

Rectangle

The Rectangle has Object (X Position, Y Position, Width, Height), Basic Attribute (Foreground, Style, Fill, Line Width), and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. The Rectangle is drawn on the display drawing area and is not a widget.

Related Display

The Related Display has Object (X Position, Y Position, Width, Height) and Visual attributes. In addition, there are Foreground, Background, Label, and Label/Name/Args attributes. The Foreground and Background colors are changed by clicking to bring up the Color Palette. The Label is the optional label that appears on the button. Normally, whether there is a label or not, there are two overlapping squares on the button to denote a Related Display control. However, if the label text begins with "-" (for example "-My Label"), the overlapping squares will not appear. The Display Label, Display Files, Arguments, and Policy are specified for each related display in the Label/Name/Args dialog box. There currently can be up to eight related displays for each button.

The Related Display provides a means of bringing up more displays. It is a menu button with a graphic consisting of two overlapping squares and optionally a label. The menu items denote the related displays that can be displayed. If there is only one related display specified in the menu, then the two overlapping squares and/or label appear in the middle of the button, and the related display is activated immediately without bringing up a menu. If there is more than one item on the menu, the squares and/or label are at the left of the button. If Ctrl-Btn1 is used in place of Btn1 to select the new display, then the parent display goes away and is replaced by the new display. The new display can also be configured to always replace the parent.

In the EDIT-mode dialog box for specifying the menu items there are four columns. The first column, the Display Label, is the label on the menu item in the menu that is brought up when the Related Display button is pressed. The second column, the Display File, is the name of the ADL file that will be displayed when the menu item with that label is selected. The third column, the Arguments, is macro substitutions that will be used as if the file were specified on the command line with a -macro argument. For example, the string:

```
SECTOR = S10, BPM=P2
```

will cause all occurrences of $(SECTOR) in the file to be replaced with S10 and all occurrences of $(BPM) with P2. Finally, the fourth column, the Policy, specifies whether to remove the parent display or not when the new one is activated. (The parent display is the one that contains the Related Display button.) The parent display will be replaced independently of the Policy if Ctrl-Btn1 is used instead of Btn1. Rows in the dialog box with blanks are ignored, even if they are before or between non-blank rows.

Neither labels or display labels cannot contain double quotes (").

Hidden buttons are implemented by setting the Visual attribute of the Related Display to invisible and placing a graphic over the related display. In this case there should only be one item in the list of displays. The graphic should make the operation of the related display clear. It should be placed above the Related Display in the stacking order and be the same size as the related display. In this case when the user clicks on the graphic, the Related Display will be executed. The Related Display is drawn as a stippled pattern to identify it. If the graphic is smaller, the stipple will show and clicking on the graphic will execute the graphic rather than the related display. If the graphic is larger, it will be possible to click on parts of the graphic without executing the Related Display. Note that when two objects of the same size are on top of each other, then the outlines shown when doing dragging, pasting, *etc.* in EDIT mode will cancel, making the objects appear to not be included. Also in EDIT mode clicking on the two objects in the proper relative positions will select the Related Display. To select the graphic, you can use Ctrl-Btn1. (See Mouse Editing Operations.) Since users are used to seeing the overlapping rectangles represent a related display, some thought should be used before implementing graphics to replace them.

The logic that is used to locate the specified files is described under ADL Files.

Scale Monitor

Also known as an Indicator. The scale monitor has Object (X Position, Y Position, Width, Height), Monitor (Readback Channel, Foreground, Background), Limits (Low Limit, High Limit, Precision), Label, Direction, and Color Mode attributes. The Scale Monitor displays the value of the process variable on a scale. The limits of the scale are the HOPR and LOPR values for the record associated with the process variable by default but may be set via the PV Limits Dialog Box.

Shell Command

The Shell Command has Object (X Position, Y Position, Width, Height) attributes. In addition, there are Foreground, Background, and Label/Cmd/Args attributes. The Foreground and Background colors are changed by clicking to bring up the Color Palette. The Command Label, Command, and Arguments are specified for each related command in the Label/Cmd/Args dialog box. There currently can be up to eight commands for each button.

The Command Label is the label on the menu item in the menu that is brought up when the Shell Command button is pressed. The string in Command will be concatenated with the string in Arguments with a space in between, and the resulting string will be executed as a system command when the menu item with that label is selected. (The Arguments are actually superfluous, since any command arguments can just as well be included in Command.) MEDM will block until the command is executed, so it is almost always wise to include an "&" at the end of the Command (or the Arguments if they re used) so that it will execute in the background.

If the command contains a "?", then the rest of the command will be ignored, and a dialog will prompt the user to complete (or otherwise edit) the command.

The command can also contain any of the special characters that are available for the Execute Menu, however, it is suggested that "?" rather than "&?" be used.

Slider

Also known as Valuator. The Slider has Object (X Position, Y Position, Width, Height), Control (Control Channel, Foreground, Background), Limits (Low Limit, High Limit, Precision), Label, Direction, and Color Mode attributes. There is also a Precision attribute. The Precision attribute is different from that in the Limits Attribute and specifies by how many units the slider moves when using Btn1 or the arrow keys as described below. It does not determine the precision for the value or limit numbers when these are shown on the slider. These are determined by the Limits Attribute.

Dragging the Slider button with Btn1 transmits values. The sensitivity with which values can be selected this way depends on the range of the slider and not on the Precision. Clicking Btn1 in the space on either side of the slider button increments or decrements the value by an amount equal to the Precision. Clicking Ctrl-Btn1 in the space on either side increments or decrements the value by an amount equal to 10 times the Precision

In addition, the arrow keys increment or decrement the value by an amount equal to the Precision, and the Ctrl-Arrow keys increment or decrement the value by an amount equal to 10 times the Precision. The left and right arrow keys are operational when the Direction is right, and the up and down arrow keys are operational when the Direction is up. The arrow keys only function when the focus is on the slider button or the space on either side of it.

In EXECUTE mode the specified precision for motion of the valuator can be changed via a dialog box which is popped up by depressing Btn3 on the Slider object. A row of toggle buttons at the top sets the Precision as powers of 10 of the numbers on the buttons. The Precision can also be set in the entry box below the buttons. In addition, the value can be set directly in the lower entry box, providing an alternative to using Btn1 and the arrow keys.

Strip Chart

The Strip Chart has Object (X Position, Y Position, Width, Height), Plot (Title, X Label, Y Label, Foreground, Background), and Units attributes. In addition, there are Period and Units attributes. The period is the time between updates. The Units attributes are set in a dialog box. For each pen in the Strip Chart there is a Channel (process variable name) and a Color which can be specified in the entry boxes in the dialog. There currently can be up to 8 pens.

Text

The Text object has Object (X Position, Y Position, Width, Height), Basic Attribute (Foreground), and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D), and Alignment attributes. In addition, there is a Text attribute, which is the text to be displayed. This text can have up to 255 characters. The Text object is drawn on the display drawing area and is not a widget. The text may extend beyond the height and width specified for the Text object or may not use up all the specified height and width. The Text can be sized as well as possible to the actual size of the text by using the Size option on the Edit Menu. This helps when trying to align the Text object with other objects.

The Text object may be created in two ways. You can drag a rectangle to the desired size, the same as for most other objects, or you can click and release Btn1 on the display drawing area. If you click and release Btn1, then the cursor changes to an I-beam and you can type the text directly onto the display rather than entering it in the Resource Palette. The text is finished when one of the following happens: (1) A button is pressed; (2) Return is typed; or (3) The pointer leaves the drawing area (either leaves the display or goes on top of a widget in the drawing area). The size of the text depends on the size of the previous object (not necessarily a Text object) in the Resource Palette. If the height is larger than 40, then 40 is used. The color is the same as the color of the previous object. You can control the size and color by first selecting an object (most usefully a Text object) with the desired size and color, then creating the new Text object by clicking and releasing Btn1.

The font used depends on the height of the Text object using an algorithm that is internal to MEDM. This algorithm may not be the most sensible, but it has been kept to avoid changing the layout of old displays. You adjust the font by adjusting the height, usually by experimentation. The text typically does not fill all of the specified height for the Text object. The text cannot contain double quotes (").

Text Monitor

Also known as Text Update. The Text Update has Object (X Position, Y Position, Width, Height), Monitor (Readback Channel, Foreground, Background), Limits (Precision), Alignment, Format, and Color Mode attributes. The Text Update is drawn on the display drawing area and is not a widget. It displays the value of the process variable in the chosen format.

Text Entry

The Text Entry has Object (X Position, Y Position, Width, Height), Control (Control Channel, Foreground, Background), Limits (Precision), Format, and Color Mode attributes. The Text Entry provides a means of displaying the value of a process variable in an entry box, where it can be edited and changed. The value is not changed until Return is pressed with the focus in the entry box. If the focus leaves the entry box, the value of the process variable is not changed and the current value is restored into the entry box.

The Text Entry in MEDM is a valid drop site only if its Format is a string. The reason is to prevent trashing the value when the process variable name for the entry is inadvertently dropped on the entry.

Composite Objects

Composite objects are created by grouping other objects via commands on the Edit menu. For many operations they are treated as one object.

The Composite has Object (X Position, Y Position, Width, Height) and Dynamic Attribute (Color Mode, Visibility, Visibility Calc, Channels A-D) attributes. Color Mode in the Dynamic Attribute is not relevant and is not used. In addition there is a Composite File attribute.

MEDM is designed so that when you change the foreground or background colors of a Composite object, you change the foreground or background colors of all the objects in the Composite simultaneously. Therefore, in order to change the colors of a lot of objects at the same time, it is only necessary to group them first, making them Composite, and then change the color via the Resource Palette.

After a Composite object has been created, it can be made to get its objects from an ADL file, rather than use the ones it currently contains. It just necessary to enter the name of a valid ADL file for the Composite File attribute. The Composite will ignore things like the display and colormap in the ADL file and just use the objects in it. The positions of the objects will be adjusted so the upper left corner of the bounding box of all of them is at the specified x and y values of the Composite. The specified height and width of the Composite are adjusted to those of the bounding box. This feature allows designing a complicated Composite independently of its role in the current display without the need to group and ungroup the objects in order to edit them. This is especially useful if there are overlapping composites.

Since the Composite has a Dynamic Attribute, its visibility may be made to depend on a CALC expression involving up to four process variables or the other visibility features of a Dynamic Attribute. When the Composite is not visible, all of its objects are hidden. The objects however continue to receive updates from Channel Access if they were ever visible. This feature can be used to make parts of a display only appear under certain conditions. More generally, by using overlapped Composites, whole sections of a display can be made to appear, disappear, or change from one thing to another depending on the conditions. This is a powerful feature that has the potential to be abused. Having the display change may be confusing and disconcerting to the user. It also uses more resources to update Composites in addition to the objects they contain, in part because the Composite and its objects are always overlapped. It uses even more resources when Composites are overlapped.

A Composite may consist of only a single object. By enclosing any object in a Composite, it can be made to have a Dynamic Attribute.

In MEDM for reasons of efficiency, only the parts of a display that need to be refreshed are actually refreshed. If the visibility of a Composite changes, the display may be updated only within the bounds of the Composite. Since the text in a Text object may extend beyond the bounds of the object, you should insure that such text does not also extend beyond the Composite boundaries, which are determined by the size of the Text object, not the size of its text. Otherwise, the visibility of the extended part of the text may not be handled correctly.

Attributes

Object

All objects have Object attributes.

X Position	x coordinate of the top left corner of the object relative to the display.
Y Position	y coordinate of the top left corner of the object relative to the display.
Width	Width of the object.
Height	Height of the object.

Basic

For the Arc, Oval, Polygon, Polyline, Rectangle, and Text.

Foreground	Foreground color. Click to bring up the Color Palette to choose the color.
Style	Edge style for the object. See Edge Style.
Fill	Fill style for the object. See Fill Style.
Line Width	Line width in pixels.

Dynamic

For the Arc, Oval, Polygon, Polyline, Rectangle, Text, and Composite.

Color Mode	Color Mode for the object. See the Color Mode attribute.
Visibility	Visibility mode for the object. See the Visibility attribute.
Color Rule	The color rule number for the object. (Integer number.) See Color Rules.
Visibility Calc	A CALC expression that determines whether the object is displayed or not. The expression should return 0 for False and anything else for True. See CALC Expression.
Channel A	Name of the main process variable associated with the object. An A in the CALC expression is replaced by the value of this process variable, and the values corresponding to G through L are obtained from this process variable.
Channel B	Name of the second process variable associated with the object. A B in the CALC expression is replaced by the value of this process variable.
Channel C	Name of the third process variable associated with the object. A C in the CALC expression is replaced by the value of this process variable.
Channel D	Name of the fourth process variable associated with the object. A D in the CALC expression is replaced by the value of this process variable.

Monitor

For the Bar, Byte, Meter, Scale Monitor, and Text Monitor.

Readback Channel	Name of the process variable to be read by the object.
Foreground	Foreground color. Click to bring up the Color Palette to choose the color.
Background	Background color. Click to bring up the Color Palette to choose the color.

Control

For the Choice Button, Menu, Message Button, Slider, and Text Entry.

Control Channel	Name of the process variable to be controlled by the object.
Foreground	Foreground color. Click to bring up the Color Palette to choose the color.
Background	Background color. Click to bring up the Color Palette to choose the color.

Limits

For the Meter, Bar Monitor, Scale Monitor, Slider, Text Monitor, and Text Entry. See the PV Limits Dialog Box for how these are used and specified.

Low Limit Source	Source for the low limit value (Channel, Default, User Specified)
Low Limit Channel Value	Value when source is Channel. (Floating point number.)
Low Limit Default Value	Value when source is Default. (Floating point number.)
Low Limit User Value	Value when source is User Specified. (Floating point number.)
High Limit Source	Source for the high limit value (Channel, Default, User Specified)
High Limit Channel Value	Value when source is Channel. (Floating point number.)
High Limit Default Value	Value when source is Default. (Floating point number.)
High Limit User Value	Value when source is User Specified. (Floating point number.)
Precision Source	Source for the precision value (Channel, Default, User Specified)
Precision Channel Value	Value when source is Channel. (Floating point number.)
Precision Default Value	Value when source is Default. (Floating point number.)
Precision User Value	Value when source is User Specified. (Floating point number.)

Plot

For the Cartesian Plot and Strip Chart.

Title	Title for the plot object.
X Label	X-axis label for the plot object.
Y Label	Y-axis label for the plot object.
Foreground	Foreground plot color. Click to bring up the Color Palette to choose the color.
Background	Background plot color. Click to bring up the Color Palette to choose the color.
Package	(Not Implemented.)

Plot Axis

For the Cartesian Plot.

Axis Style	Axis style for the plot axis. See Cartesian Plot Axis Style.
Axis Range	Source of the range for the plot axis. See Cartesian Plot Range Style.
Minimum Value	Minimum value shown on the plot axis. (Floating point number.)
Maximum Value	Maximum value shown on the plot axis. (Floating point number.)
Time Format	Time format for the axis. See Time Format.

Cartesian Plot Axis Style

For the Cartesian Plot.

linear	Use a linear axis.
log10	Use a log axis.
time	Use a time axis.

Cartesian Plot Range Style

For the Cartesian Plot.

from channel	Get the axis range from the process variable. (Usually LOPR and HOPR fields.) The range returnec by the record support may not be appropriate for all fields of the record, may not be appropriate for the current values of the process variable, or LOPR and HOPR may not have been set to the best values.
user-specified	Specifically specify the minimum and maximum values for the axis.
auto-scale	Let the graph routine decide on the axis range depending on the data. This usually gives an attractiv graph, but may cause the graph to flash if the range changes frequently.

Cartesian Plot Style

For the Cartesian Plot.

point	Plot the data as points.
line	Plot the data as lines.
fill-under	Plot the data as lines which are filled under (or over) from the line to the axis.

Cartesian Plot Time Format

For the Cartesian Plot when there is a time axis.

hh:mm:ss	*E.g. 22:05:10, 05:43:35*
hh:mm	*E.g. 22:05, 05:44*
hh:00	*E.g. 22:00, 06:00*
MMM DD YYY	*E.g. Jun 27, 1997*
MMM DD	*E.g. Jun 27*
MMM DD hh:00	*E.g. Jun 27 22:00*
wd hh: 00	*E.g. Tue 22:00*

Color Mode

For the Bar, Byte, Choice Button, Menu, Message Button, Meter, Scale Monitor, Slider, Text Entry, Text Monitor, and all objects with a Dynamic attribute.

static	Show the object in its normal colors.
alarm	Show the object in alarm colors based on the severity of the associated process variable. The alarm colors are Green for NO_ALARM, Yellow for MINOR_ALARM, Red for MAJOR_ALARM, White for INVALID_ALARM, and Gray if the alarm has an unknown value (because of an error ir the record, for example).
discrete	If color rules are implemented and the object has a Dynamic attribute, use the color rules. See Colo Rules. Otherwise the same as static.

Color Rule

This is part of the Dynamic attribute that is used if color rules are implemented. See Color Rules.

set #1	Use color rules defined by set #1.
set #2	Use color rules defined by set #2.
set #3	Use color rules defined by set #3.
set #4	Use color rules defined by set #4.

Direction

For the Bar Monitor, Byte Monitor, Scale Monitor, and Slider.

up	Direction for the object is up or up/down.
right	Direction for the object is right or left/right.
down	No longer used. In ADL files MEDM treats this the same up for backward compatibility.
left	No longer used. In ADL files MEDM treats this the same right for backward compatibility.

Edge Style

This is part of the Basic attribute.

solid	Use solid lines.
dash	Use dashed lines.

Erase Mode

For the Cartesian Plot with an erase channel.

if not zero	Erase the plot if the erase-channel process variable is not zero.
if zero	Erase the plot if the erase-channel process variable is zero.

Fill Mode

For the Bar Monitor.

from edge	Fill from the lower end to the current value.
from center	Fill from zero to the current value.

Fill Style

This is part of the Basic attribute.

solid	Fill the whole shape with color.
outline	Only color the outline.

Image Type

For the Image.

no image	Do not display the image.
gif	The image is Graphics Interchange Format (GIF), the only allowable image type at this time.

Label

For the Meter, Bar Monitor, Scale Monitor, and Slider.

none	No extra features, except the limits are shown for the meter. You can eliminate the limits on the meter by resizing it vertically so they do not show.
no decorations	Same as none except for the Bar Monitor. For the Bar Monitor only the background and the bar show. This allows making bar graphs in MEDM.
outline	Show limits.
limits	Show limits and a box for the value (except there is no box for the Slider).
channel	Show limits, a box for the value, and the process variable name (except there is no box for the Slider).

Plot Mode

For the Cartesian Plot with a scalar process variable.

plot n pts & stop	Plot n points corresponding to the first n changes of the process variable, then do not plot any more points.
plot last n pts	Plot n points corresponding to the last n changes of the process variable, overwriting previous points.

Related Display Mode

For the Related Display.

create new display	Create a new display and leave the current one.
replace display	Create a new display that replaces the current one.

Related Display Visual

For the Related Display.

menu	Use a pull down menu for the choices.
a row of buttons	Use a row of buttons for the choices.
a column of buttons	Use a column of buttons for the choices.
invisible	Do not show anything for the choices. This mode is intended to be used with a graphic on top of th related display. The graphic should make the operation of the Related Display clear.

Stacking

For the Choice Button.

column	The buttons are arranged in a row. (This appears to be a mistake, which will not be corrected because of existing screens.)
row	The buttons are arranged in a column. (This appears to be a mistake, which will not be corrected because of existing screens.)
row column	The buttons are automatically arranged in rows and columns.

Text Align

For the Text and Text Monitor.

horiz. left	Align the text at the top left of the object. (Capital letters will line up with the top of the object, and the text will start at the left of the object.)
horiz. centered	Align the text at the top center of the object. (Capital letters will line up with the top of the object, and the text will be horizontally centered in the object.)
horiz. right	Align the text at the top right of the object. (Capital letters will line up with the top of the object, ar the text will end at the right of the object.)
vert. top	No longer used. In ADL files MEDM treats this the same as horiz. left (as it used to) for backward compatibility.
vert. centered	No longer used. In ADL files MEDM treats this the same as horiz. center (as it used to) for backward compatibility.
vert. bottom	No longer used. In ADL files MEDM treats this the same as horiz. right (as it used to) for backwar compatibility.

Text Format

For the Text Entry and Text Monitor. For all of the formats, the result depends on the number itself and the precision as returned from channel access for the process variable. The precision is typically the PREC field for the associated record and may be changed by the PV Limits Dialog Box. The PREC field is only determined when the process variable is connected, so subsequent changes to the PREC do not affect the format until MEDM is restarted.

decimal	The text is a number with or without a decimal point, but not in exponential notation. *E.g.*10000.00 or 10000. Very large numbers, which do not fit in the allocated space for text, are converted to exponential regardless. It is what is returned by the EPICS routine cvtDoubleToString().
exponential	The text is in exponential notation. *E.g.* 1.00e+04.
eng. notation	The text is in engineering notation. *E.g.* 10.00e+03.
compact	The text is supposedly in the most compact form of decimal or exponential. (Similar to the %g format in C.) The form may not be the most compact, however. It is what is returned by the EPICS routine cvtDoubleToCompactString().
truncated	The text is truncated to the largest integer. *E.g.* 10000.
hexidecimal	The text is truncated to the nearest integer and shown in hexidecimal. *E.g.* 0x3e8.
octal	The text is truncated to the nearest integer and shown in hexidecimal. *E.g.* 1750.
string	This is the same as decimal except that for large numbers or precision, it can be in exponential format. It is what is returned by the EPICS routine cvtDoubleToString().

Time Units

For the Strip Chart.

milli-second	The update period is in milli-seconds.
second	The update period is in seconds.
minute	The update period is in minutes.

Visibility Mode

This is part of the Dynamic attribute.

static	The object is displayed always. Updating objects are always above non-updating objects. Specifyin a process variable in the Dynamic Attribute makes the object update and insures that it will retain it stacking order among other updating objects even if it never changes its state and is always displayed. If retaining stacking is not necessary, it is more efficient to not specify a process variabl if you always want it to display the object.
if not zero	The object is displayed if the process variable is not zero.
if zero	The object is displayed if the object is zero.

CALC Expressions

CALC expressions are used to determine visibility for objects with a Dynamic Attribute and to determine the frame number for the Image, which also has a Dynamic Attribute. For visibility the expression should return 0 for False and anything else for True. For the frame number for the Image it should return a number, which will be rounded to the nearest integer.

The complete syntax of the CALC expression is described in detail in the Record Reference Manual found in the Epics Documentation. It is a case-independent C expression that can include functions such as ABS, SQR, MIN, MAX, CEIL, FLOOR, LOG, LOGE, EXP, SIN, SINH, ASIN, COS, COSH, ACOS, TAN, TANH, and ATAN, as well as the usual C operators, except that != is replaced by # and == is replaced by =. The letters A-L obtain their values from the Dynamic Attribute’s Channels A-D and are replaced as follows.

A	The value of Channel A.
B	The value of Channel B.
C	The value of Channel C.
D	The value of Channel D.
E	Reserved.
F	Reserved.
G	The COUNT of Channel A.
H	The HOPR of Channel A.
I	The STATUS of Channel A.
J	The SEVERITY of Channel A.
K	The PRECISION of Channel A.
L	The LOPR of Channel A.

Examples are:

Show the object whenever the value of Channel A is zero (The same as a Visibility of "if zero"):

```
!A
```

Show the object whenever the value of Channel A is not zero (The same as a Visibility of "if not zero"):

```
A
```

Show the object whenever the value of Channel A is 12:

```
A=12
```

Show the object whenever the value of Channel A is not 12:

```
A#12
```

Show the object whenever the values of Channel A, Channel B, and Channel C are all negative:

```
A<0&&B<0&&C<0
```

Show the object whenever the main process variable is within 90% of its HOPR or LOPR limits:

```
A<.9*L||A>.9*H
```

Show the object whenever the SEVERITY of the main process variable is not 0:

```
!J
```

Grid

There is a grid in EDIT mode that can be toggled on and off. Each display has its own grid and grid properties. These are saved in the ADL file. The grid properties (the grid spacing, whether the grid shows, and whether snap to grid is in effect) for the current display can be set via the Grid options on the Edit menu or on the Resource Palette for the display. The grid is drawn in the foreground color of the display. This color can be changed in the Resource Palette. The Space Evenly options on the Edit menu use the current grid spacing, which can be changed to allow different spacing when using these options.

When snap to grid is enabled, moving with Btn2 (but not the arrow keys), pasting, and object creation constrain the upper left corner of the (group of) selected objects to be at a grid point. For object creation all corners are constrained to be at grid points. These operations do not change the locations of selected objects relative to each other. Objects can also be aligned to the grid via the Edit menu. The Edit-menu align options align each selected object, and may change the locations of selected objects relative to each other.

Macro Substitution

MEDM now supports macro substitution (including nested or parent-to-child substitution) for related displays.

Related displays can be called with an arguments string of the form "name=value [,name=value]?". The related display then substitutes in its space all occurrences of "$(name)" with "value".

A parent related display can pass to it's child related display a value in its space by passing an argument string of the form "name=$(name)XYZ", etc.

Shell command and argument strings are also substituted in related displays, and referenced similarly via the $(name) construct.

Drag-And-Drop

MEDM has a drag-and-drop mechanism in EXECUTE mode, using Btn2. This allows dragging the process variable name(s) associated with an MEDM object to another place. The place must be a valid Motif drag-and-drop site. Such places include most Motif text fields. Programs like the knob manager, KM, which is an EPICS extension, have been designed to have valid Motif drop sites, and it is easy to drag a process variable name from an MEDM display to KM and have it be attached to a knob.

Pressing the Escape key will abort the drag and drop. There will be a do-not-enter symbol (circle with a bar across it) when the drag is over an invalid drop site and an arrow when the drop site is valid. There is also always a symbol composed of two overlapping sheets of paper that indicate this is a copy operation (the only type available in MEDM). When dropping the process variable names, they will appear to go back to where they originated if the drop site is invalid and will appear to be absorbed by the drop site otherwise.

The color of the names depends on the alarm severity. Green is NO_ALARM; Yellow is MINOR_ALARM; Red is MAJOR_ALARM; White is INVALID_ALARM; and Gray means the alarm has an unknown value (owing to an error in the record, for example).

The valid drop places unfortunately do not include XTerm?s, and the Motif Btn2 drag-and-drop is not the same as the common Btn2 paste associated with the clipboard. To help overcome this problem, the EPICS extension, Namecapture, is a program designed to help with transferring names obtained by Motif drag-and-drop to other programs. It automatically converts a dropped name to an X selection so the common Btn2 paste works. There are also ways in Namecapture to add the dropped names to a file, even to generate lines in a BURT request file. Both Namecapture and KM are described in the online EPICS documentation.

It is not necessary to drop the name in a valid place; it can be read and discarded. Consider using the PV Info feature to look at (as opposed to dragging and dropping) process variable names.

The Text Entry in MEDM is a valid drop site only if its Format is a string. The reason is to prevent trashing the value when the process variable name for the entry is inadvertently dropped on the entry.

For the Cartesian Plot, the names are in columns corresponding to the traces. The last row corresponds to the trigger and erase channels, whether they are defined or not.

Dialog Boxes

Display List Dialog Box

The Display List Dialog Box is invoked from the Execute-Mode Menu. It has a list of the current displays. You can select one or more displays. There are buttons to Raise or Close the selected displays and to Close, Refresh, or get Help for the dialog box itself. Displays that are raised should appear on the current Workspace. Displays that are closed, whether from this dialog box or from the Exeute-Mode Popup Menu, must be reopened from the File menu. The dialog box should automatically refresh without using the Refresh button.

PV Limits Dialog Box

The PV Limits dialog box may be invoked from the Execute-Mode Menu in EXECUTE mode and from the Channel Limits button in the Resource Palette in EDIT mode. When invoked from the Execute-Mode Menu, you are prompted with a PV cursor to pick an object on the display.

The limits of an object are the endpoints of the range of values that is displayed. For example, a Meter has a maximum and a minimum value, and values beyond these limits are not displayed correctly. The Meter becomes "pinned". The precision is the number of digits displayed after the decimal point. The limits and the precision of numbers displayed in an object in can be specified at design time and changed at run time for all objects for which this is relevant. This is accomplished through the PV Limits Dialog Box. This dialog box looks the same in EDIT as in EXECUTE mode, but the meaning and the available options may be different. In all cases, options that are not meaningful are not selectable and appear grayed out.

In EXECUTE mode the process variable name will appear at the top of the dialog box. IN EDIT mode it will just have EDIT Mode Limits at the top.

There are three items that can be set in the dialog box: the Low Limit, the High Limit, and the Precision. For each of these items, the value used for the item may come from three different sources: Channel, Default, and User Specified.

Channel

In either EXECUTE or EDIT mode specifying Channel means the values in EXECUTE mode come from the record associated with the process variable. This is the default and formerly was the only option available in MEDM. Typically, the High Limit comes from the HOPR field, the Low Limit comes from the LOPR field, and the precision comes from the PREC field. More accurately, they are whatever is returned from the record-support routine, get_graphic_double(), when Channel Access asks for these limits and precision. This depends on the record in question and on which field in the record. Fields other than the VAL field often just return the default value zero for all three items, even when HOPR, LOPR, and PREC are returned for the VAL field. Some records, especially homegrown ones, may not even have these fields and may not return anything sensible. Moreover, if the person who designed the database did not specifically specify values for the HOPR, LOPR, and PREC fields (a not uncommon occurrence), the limits and/or precision will default to zero. In most cases, however, the database designer has specified the values to be ones appropriate for use in applications like MEDM. That is why these fields are there in the first place. These values are fixed in the record, and they apply to all objects using that process variable in all MEDM screens in all running MEDMs. If both the High Limit and the Low Limit are zero, MEDM converts the High Limit to 1.0.

Default

In EXECUTE mode specifying Default means the values used are those specified by the designer of the MEDM screen or the MEDM defaults if none were specified. The MEDM defaults are 0.0 for the High Limit, 1.0 for the Low Limit, and 0 for the precision. The default value cannot be changed in EXECUTE mode. Use User Specified instead. In EDIT mode the screen designer specifies the default values via the text entry box and by picking this source option specifies that these values will be used unless the user specifies something different. The screen designer can thus override what the database designer specified (or failed to specify).

User Specified

This option is only available in EXECUTE mode. The user of the screen can set the values to whatever he or she wants. It may be appropriate, for example, to set a reduced range for an object when doing fine-tuning. The user can override both the screen designer and the database designer by using this option and entering new values. He can revert to either of the other sources by choosing those options.

Both limits and precision are relevant and may be specified for the Meter, Bar Monitor, Scale Monitor, and Slider. Only the precision is relevant and may be set for the Text Monitor and Text Entry. None of these is relevant for other MEDM objects and the dialog box should not be available, nor should there be limits attributes stored in the ADL file. The Cartesian Plot has other ways to set its limits. Currently, there is no way to set the limits for the Strip Chart.

Message Window Dialog Box

MEDM prints most error and information messages to both stderr and the MEDM Message Window. This window will be created when the first message is written. By default it will be raised when each important new message is posted. (Some messages are for logging purposes only and will not cause the window to raise.) When it is raised the window should appear in the current workspace of a multi-workspace window manager, no matter whether it previously was in another workspace or previously was minimized. It should be at its last former position, so you can place it where you find convenient. The window can be set to not automatically raise by setting the toggle button labeled "Raise Message Window When Important Message is Posted" at the bottom of the window to off. If the window is closed or otherwise not visible, it can be brought up via the View menu on the MEDM main window.

There are buttons to Close, Clear, Print, Mail, and get Help. If part of the text in the window is selected, both Print and Mail will only use the selection. Otherwise, the whole window is used. Print uses the environment variable PSPRINTER, which must be set. You can change (or set) the name of the printer via the Printer Setup item in the File menu. You can also select all or part of the window and use the usual X-windows cut and paste to move the text to another program or file. Print and Mail are not available on WIN32.

The initial position of the Message Window can be changed by the following resources (shown with typical values):

```
Medm.errorMsgS.x: 10
Medm.errorMsgS.y: 10
```

You could put these lines in your .Xdefaults file, for example.

Statistics Window Dialog Box

The Statistics Window can be accessed from the View menu in the MEDM main window. It shows statistics about various things that are happening in MEDM. These statistics are accumulated by a routine that is called by a timer set to 1 second. The actual elapsed time between calls may be slightly different, depending on the load in MEDM. There are Close, Reset, and Mode buttons at the bottom of the dialog box. The Close button closes the dialog box. The reset button resets the start time and the accumulated values to zero. The mode button toggles between the two modes of displaying statistics, either for the last interval or for the accumulated averages.

Last Interval Mode

Time Interval: The actual time for the last interval.

CA Channels: The number of channels (one per process variable) requested by all the objects in all the screens.

CA Channels Connected: The number of channels successfully connected.

CA Incoming Events: The number of events received from Channel Access in the last interval. An event is received for a channel when:

- A significant change in the connection occurs (*e.g.* connect, disconnect, reconnect).
- There is a significant change in the channel (*e.g.* value, alarm status, severity status).
- A requested value is returned.
- There is a change in access rights (read, write).

MEDM Objects Updating: The number of objects that could potentially be updated (redrawn on the screen). (An MEDM object can have more than one process variable associated with it.)

MEDM Objects Updated: The number of objects actually updated in the last interval. (Can be larger than the number updating if some were updated more than once).

Update Requests: Number of object updates requests that were queued for processing.

Update Requests Discarded: Number of objects that got a new update request before the old one was processed.

Accumulated Averages Mode

CA Incoming Events: Average of CA Incoming Events (see above) over the time elapsed since the last reset.

MEDM Objects Updated: Average of MEDM Objects Updating.

Update Requests: Average of Update Requests.

Update Requests Discarded: Average of Update Requests Discarded.

Total Time Elapsed: The time over which the averages are calculated.

PV Info Dialog Box

The PV Info dialog box is invoked from the Execute-Mode Menu. You are prompted with a PV cursor to pick an object on the display. The dialog box then shows information about all the process variables associated with that object. For each process variable the information includes the name, the description, the record type (AO, AI, CALC, *etc.*), the native type (DBF_DOUBLE, DBF_STRING, DBF_ENUM, *etc.*), the count (length of the array or 1 if not an array), the access rights (RW), the IOC, the current value with associated timestamp, and its alarm status, as well as other information depending on the native type. Only the value of the first element of an array is shown. The record type is a newer feature of Channel Access and may not be available if the version running in the IOC is older. The information shown is that stored by MEDM, except for the current value and the associated time stamp, which are obtained when you pick the object. Items such as HOPR and LOPR are the startup values and are not updated automatically except when the connection changes. It should be possible to get some of this information if the process variable has lost its connection; that is, when the object is white but was previously not white.

When choosing an object with the PV cursor, the smallest object under the cursor will be chosen. If the object selected by the PV cursor is part of a Composite that has process variables associated with it, then the dialog box will show the process variables for the object if it has any. Otherwise it will show the process variables for the Composite. If an object that is part of a Composite is hidden because the Composite is currently not visible, then the process variables for the Composite will be shown. If the object is not part of a Composite and is not currently visible, the dialog box will still show its process variables. The name of the MEDM object is shown in the dialog box to help avoid confusion in ambiguous cases.

When running through some older versions of the Gateway, there may be a long delay (1 minute) before the dialog box comes up. This is owing to a bug in the portable Channel Access server, which has now been fixed.

Help

Help from the menus and context sensitive help uses Netscape. If Netscape is not up, it takes a little while for it to come up after the first help request; however, once it is up, new help topics appear quickly inside the existing Netscape. The Netscape window raises so as to be visible whenever help is called. There is a dialog box for Help on Help (in case Netscape is not working) describing the Netscape help and what is necessary to make it work. The name for the Netscape executable can be specified in the environment variable NETSCAPEPATH.

There is bubble help to identify the objects in the Object Palette, and there is a summary of EDIT operations on the Edit menu, which appears on the main window and as a popup menu when editing a display.

Each display has a provision for help on the display. This help is implemented via the Help option on the window menu button (the button with a short bar on it located at the left of the title bar for the display). This button does not exist unless the environment variable MEDM_HELP is defined. In that case when this option is selected from the window menu, MEDM makes a system call of the form "$MEDM_HELP full-adl-pathname &". The value of MEDM_HELP should be the name of a program that uses the full pathname of the ADL file for the display to provide help, for example, by bring up Netscape with an appropriate page based on the filename. What the program does or how it is written is independent of MEDM.

Medm Smart Startup

In order to amortize the cost of Xt Intrinsics and Motif initialization over many displays, and to offer an efficient way to start up MEDM displays "after the fact", an MEDM remote request protocol has been established.

If MEDM is started with the "-attach" option on the command line, it looks for another MEDM running (on any host machine) which has the same X Windows display as the requested display. (Note that the X Windows display refers to the environment variable DISPLAY or a display specified on the command line as *e.g.* "-display phoebus:0", which is different from an MEDM display as specified by an ADL file.) If one is found, the MEDM display request is forwarded to the remote MEDM for processing.

What is common among all attached MEDMs is that they were started from the same X display. They may have been started on different machines on different networks by different users. Note that the user and access permissions will be the same as for the original, host MEDM. This may lead to confusing behavior if successive MEDMs are started from different machines, from different networks, or by different users.

The default behavior, equivalent to "-local" on the command-line, is to not participate in the smart-startup protocol and to act independently of other MEDMs. Also, you can use the "-cleanup" parameter on the command line to ignore any existing MEDMs and to use this MEDM as the remote MEDM for future remote requests.

MEDM implements the smart startup by setting a property on the root window of the display. This property stores the X window id of the MEDM that is available for remote requests. An MEDM using "-attach" looks for the property when it starts up. If it finds the property, it gets the X id from the property and begins sending all of the information from its command line to the host MEDM by generating ClientMessage events. These messages are passed on the display, not on the machines where the MEDMs may be running. The host MEDM processes these events along with its other events, such as ButtonPress, for example, and starts up the requested ADL displays. After all the information has been sent the second MEDM exits.

If MEDM crashes badly or is killed with a SIGKILL, which cannot be trapped, this property may still be left on the root window. This should not cause a problem as the next MEDM started with "-attach" will find out that the window id is not valid and start itself as if "-cleanup" had been specified.

You can look at and remove the property yourself if you wish:

```
% xprop -root | grep MEDM
```

If there is one, you can remove it by doing the following:

```
% xprop -root -remove <property-name>
```

where <property-name> is the name of the property found. (The property name will have the MEDM version and have _EDIT_ or _EXEC_ in the name, depending on how the remote MEDM was started.) Most times, MEDM will clean up properly when it exits, even if it was stopped abnormally. It is no longer necessary to use "-cleanup" to clean up the root window property, and in fact it should rarely be necessary to use it at all.

Display Colors

Each display in MEDM has its own colormap with up to 65 colors. The colors of all the objects and the display itself are taken from this colormap. These colors may be seen, for example, in the color palette. The colors for each object are specified as a number, which is an index into the colormap array. If the colormap for a display is changed, the colors of the object will be different if the color for that index is different. Different displays may simultaneously have different colormaps. Of course, if there are more total colors than the X colormap can hold, some of these will not display correctly. This is particularly a problem for 8-bit X displays. It can be overcome by specifying a private X colormap for MEDM using the command-line options. The display colormap may be specified in one of two ways.

The first way is by specifying the Colormap attribute of the Display object. The Colormap property is the name of a file containing a "color map" block. The specified file may have other blocks than a "color map" block and, in fact, any ADL file containing a "color map" block is a valid choice.

The second way is by specifying a "color map" block in the ADL file. If the Colormap attribute is blank, this is done for you automatically when you save the display in an ADL file. You may use such a block as a template to define your own colors. The items in the "color map" block are the RGB values of each color index. The format should be obvious from the template.

A display created from New on the File Menu has the MEDM internal default colormap. This default is specified in the siteSpecific.h file in the MEDM distribution and may be changed if you build MEDM yourself.

When a display is saved, if there is a Colormap specified in the Display object, no "color map" block is written to the ADL file. Otherwise, the current display colormap is written as a "color map" block. Note that setting the Colormap property to blank will result in the current colormap, not the default colormap, being written to the ADL file. Specifying a colormap file saves room in the ADL file and allows using a consistent colormap for a group of displays at the expense of reading an additional file when executing the display. If the Colormap is not specified, and the colormap in the ADL file is not edited, then the colormap will be consistent for all displays created with the same MEDM. The latter has been the usual method and has a number of advantages.

There is no means in MEDM of editing the display colormap except by editing a colormap file. Keep in mind that once displays are created, changing the colormap globally may make them unaesthetic. As with the choice of fonts, the colormap strategy is one that should be made early and wisely.

Resizing Displays

Displays in MEDM do not have resize handles and may only be resized by invoking the resize method appropriate for the window manager in use. This is typically by right clicking the button on the left of the title bar.

In EXECUTE mode, the contents of the display will be scaled to fit the new size of the display. If you wish to preserve the aspect ratio of a display, you can terminate the resize option with Shift-Click instead of Click. In that case MEDM modifies the size you have chosen by decreasing the dimension that is too large. Text tends to scale more attractively if you keep the aspect ratio fixed.

In EDIT mode only selected objects are scaled to fit the new size of the display. The others are left where they were. This gives you some flexibility. There is no aspect-ratio preserving option.

Editing

When MEDM is in EDIT mode, editing operations are initiated through mouse or pointer input, keyboard input, or menu selection.

Mouse Editing Operations

Keyboard Editing Operations

Menu Editing Operations (Edit Menu)

Mouse Editing Operations

Several editing operations using the mouse are initiated by choosing one or more objects with the mouse. If the mouse is clicked, the object under the cursor is chosen. If the mouse is dragged, a rubberband box appears, and all of the objects included within the box when the button is released are chosen. Chosen objects should be distinguished from selected objects. Chosen objects are those chosen for the current operation. Selected objects are those that are highlighted. Chosen objects can be added to or removed from the selected objects or may replace them, depending on the operation.

	Select Mode (Pointing Hand Cursor)
Btn1	Vertex Edit. If a Line, Polyline, or Polygon was previously the only object selected, and if the button is pressed near a vertex, dragging moves the vertex. During dragging, when the Shift key is depressed, the dragging is constrained to be in a direction that is a multiple of 45 degrees. (The Shift key must be pressed after the button is pressed or the operation will be treated as a Shft-Btn1 operation.) Select objects. If the conditions for a vertex edit are not satisfied, then choose objects by dragging or clicking. These become the new selected objects. If only one object was previously selected, however, then additional clicking on that element toggles its selection. Clicking on the background toggles selection of the display. In the case of completely overlapping objects, only the top one can be selected in this way.
Shift-Btn1	Add or remove objects from the selected objects. Choose objects by dragging or clicking. The chosen objects are removed from or added to the selected objects, depending on whether they were included or not before.
Ctrl-Btn1	Select an object that may be under another object. Choose objects by clicking. Successive clicks will cycle the choice through any overlapped objects. The chosen object will be the new selected object. Only one object can be selected this way.
Btn2	Move objects. Choose an object by pressing the button on it, then drag the object to a new position. If more than one object was previously selected, then dragging any one of the selected objects will move all of them. In the case of completely overlapping objects, only the top one can be moved in this way. An object cannot be moved off the visible part of the display in this way. (The comparable keystroke operation does allow moving off the visible part of the display.) Btn2 supports snap to grid. The Motif Cancel key (usually Esc or Escape) cancels the move as long as the button has not been released.
Shift-Btn2	Not used.
Ctrl-Btn2	Resize objects. Choose an object by pressing the button on it, then drag the object to resize it. If more than one object was previously selected, then clicking and dragging any one of the selected objects will resize all of them. The top left corner is held fixed during resizing. In the case of completely overlapping objects, only the top one can be resized in this way. . The Motif Cancel key (usually Esc or Escape) cancels the resize as long as the button has not been released. (You can release the Crtl key before pressing Cancel, though.)
Btn3	Popup the editing menu.

	Create Mode (Crosshair Cursor)
Btn1	Create the object in the method appropriate for the object, usually by dragging to the desired size. (The Text, Polyline, and Polygon objects have special create methods.) The object must start on the visible part of the display and not on top of an object that is a widget, but it may extend past the visible part if the button is released outside the display or it may end on a widget. (It is not suggested to do this, however.) The mode will then change to Select Mode. Create supports snap to grid and aligns all corners to the grid, not just the upper left corner. The Motif Cancel key (usually Esc or Escape) cancels the create as long as the button has not been released.
Btn2	Not used.
Btn3	Popup the editing menu.

Keyboard Editing Operations

The keyboard editing operations duplicate actions that can be done by mouse operations. There is much finer control, however, with the keyboard operations

Arrow Key	Move selected objects. Objects can be moved off the visible part of the display.
Shift-Arrow Key	Same as for Arrow Key.
Ctrl-Arrow Key	Resize selected objects. The top left corner remains fixed during resizing.

Menu Editing Operations

In EDIT mode clicking Btn3 anywhere on the display brings up an editing menu. This menu has the following items. Except for the Object option, these are the same items as for the Edit menu on the main MEDM window.

Undo	Reverts to the last saved state of the display. The state is saved before most operations that would seriously change the display and also when Undo is chosen. It is not saved on keyboard editing operations. Repeated Undo?s toggle between the current state and the last state, so you can pick the one you like best.
Object	This is another way to select an object to create besides using the Object Palette. There are three pull-right submenus that group the objects into Graphics, Monitors, and Controllers.
Cut	Deletes the selected objects and saves them in an internal clipboard.
Copy	Copies the selected objects to the internal clipboard.
Paste	Pastes the objects in the internal clipboard into the display at the position of the cursor. They are selected and can be moved with the mouse until a mouse button is released. Objects are constrained to remain in the visible part of the display. The Motif Cancel key (usually Esc or Escape) cancels the paste. Paste supports snap to grid.

Raise	Raises the selected objects in the display list. Note that graphics objects are drawn on the background of the display window and will always be below the other types of objects.
Lower	Lowers the selected objects in the display list. Note that graphics objects are drawn on the background of the display window and will always be below the other types of objects.
Group	Makes a composite object of the selected objects.
Ungroup	Decomposes the selected composite object into its components.
Align	Left: Align the left edges of all selected objects. Horizontal Center: Align the centers horizontally of all selected objects. Right: Align the right edges of all selected objects. Top: Align the top edges of all selected objects. Vertical Center: Align the centers vertically edges of all selected objects. Bottom: Align the bottom edges of all selected objects. Position to Grid: Align the upper left corner of all selected objects to the grid Edges to Grid: Align the edges of all selected objects to the grid. This may both move and resize the objects.
Space Evenly	Horizontal: Make the horizontal spacing between the selected objects equal to the grid spacing. The leftmost object will not move. Best applied to a row of objects. Does not affect object size. Vertical: Make the vertical spacing between the selected objects equal to the grid spacing. The topmost object will not move. Best applied to a column of objects. Does not affect object size. 2-D: Line up an array of objects. Best applied to objects already somewhat lined up in rows and columns. The objects will be aligned as well as spaced. The top left object will be at the upper left corner of the former bounding box of all the objects. The objects will be vertically spaced according to the grid spacing and the average height, and horizontally spaced according to the grid spacing. Does not affect object size.
Center	Horizontally in Display: Center the selected objects horizontally with respect to the display. If there is more than one object selected, the group is centered. Use Align in combination with Center to center each selected object. Vertically in Display: Center the selected objects vertically with respect to the display.

Orient	Flip Horizontally: Flip the selected objects horizontally about the midpoint. Flip Vertically: Flip the selected objects vertically about the midpoint. Rotate Clockwise: Rotate the selected objects clockwise about the midpoint. Rotate Counterclockwise: Rotate the selected objects counterclockwise about the midpoint. These operations in most cases change only the location of the objects. They do not change the orientation of what is in the objects. For example, text still reads from left to right and meters are never upside down. The internal orientation of Polylines, Polygons, and Arcs does change. Note that because of roundoff, the objects may creep if these operations are replied repeatedly. This tends to happen if the height or width is not an even number of pixels. In addition, objects are constrained to have coordinates greater than or equal to zero.
Size	Same Size: Make the height and width of all the selected objects the same and equal to the average height and width of the selected objects. Text to Contents: Make the sizes of selected text objects be just large enough to fit their contents. Only the horizontal position and the width of the enclosing box are changed. The appearance and position of the text itself should not change. Other types of selected objects are ignored, so this can be done for all text objects in the display by selecting everything first. Resizing the text objects should make alignment and spacing easier.
Grid	Toggle Show Grid: Toggle whether the grid is displayed or not. (This can also be done in the Resource Palette for the Display.) Toggle Snap to Grid: Toggle whether the Snap to Grid is in effect or not. (This can also be done in the Resource Palette for the Display.) When Snap to Grid is in effect, objects created or moved with the mouse will align their upper left corners to the grid. Grid Spacing: Brings up a dialog box to specify the grid spacing. The current spacing is shown. (This can also be done in the Resource Palette for the Display.)
Unselect	Unselects all objects.
Select All	Selects all the objects in the display.
Find Outliers	Find objects that are totally or partially off the visible part of the display. (This can happen, for example, if the display is resized.) The number of such objects will be presented in a dialog box. If there are any objects, they will be selected, and there will be a button on the dialog box to list them on the MEDM Message Window. They can subsequently be moved or deleted if desired.
Refresh	Redraw the display. This will cause widgets that have been moved to be back in the correct stacking order.
Edit Summary?	Pops up a summary of pointer and key edit operations.

Execute-Mode Popup Menu

In EXECUTE mode there is a popup menu on each display that is invoked by pressing Btn3 anywhere on the background or on any object (*e.g.* the Cartesian Plot or Slider) that does not respond to Btn3 itself. This menu contains the following items:

Print	Prints the display. See Printing.
Close	Closes the display.
PV Info	Prompts you with a PV cursor to choose an object, then displays information about any process variables associated with that object. See PV Info under Dialog Boxes.
Execute	Brings up a sub menu with options that are determined by the MEDM_EXECUTE_LIST environment variable. See Execute Menu.

Printing

In EXECUTE mode the display can be printed by using the popup menu invoked by Btn3 or with the Print item on the main File Menu. For this to work the environment variable PSPRINTER must be set to a printer that accepts Postscript. You can change (or set) the name of the printer via the Printer Setup item on the File Menu. This feature is not available on WIN32.

Execute Menu

The Execute Menu, accessed by pressing Btn3 in EXECUTE mode, is a configurable menu. The configuration is specified in the MEDM_EXEC_LIST environment variable. This variable is of the form:

```
name1; program1[[:name2; program2]?].
```

The items between colons represent each menu item. All of the characters up to the first semi-colon in each item are the text that appears on the menu. The remaining characters represent the system command that will be executed when the item is selected. Note that if the command is not run in the background (using & at the end), it will hold up MEDM until it finishes. The system command can include the following special characters:

&P	The name(s) of the process variable(s). A PV cursor will appear to choose the object for which the process variable(s) are wanted. If there is more than one name, the names are space delimited.
&A	The full path name of the ADL file associated with the display.
&T	The short name of the file. (This should be the same as the title of the display.)
&?	Truncate the command at this point and prompt the user to finish or otherwise edit the command.

Examples are:

```
setenv MEDM_EXEC_LIST 'Probe;probe &P &:PV Name(s);echo &P'
setenv MEDM_EXEC_LIST 'Full Name;echo &A:Short Name;echo &T'
setenv MEDM_EXEC_LIST 'XTerm;xterm -fg black -bg white &'
setenv MEDM_EXEC_LIST 'XTerm;xterm &?'
```

Color Rules

Color Rules are a means of changing the color of an object based on a calculation. They are potentially part of the Color Rule part of the Dynamic Attribute but are not currently supported. This capability can be accomplished by an animated image, however, as described in the section on the Image.

Color Conventions

The following color conventions should be used when creating displays:

Controls

Any control button, slider, menu, related display, text entry, *etc.* that makes something happen should be BLUE (second row blue in the MEDM default color palette). This is useful because operators can quickly observe "hot spots" on a display where they can make something happen (change a value, call up another display, etc). The context or icon of "the blue thing" should adequately convey WHAT will happen if they click on it.

Text

All text updates should be displayed on a background that allows all alarm colors to be visible (if the alarm dynamic is set). We usually use light gray or dark gray (almost black). Another "RULE OF THUMB" that we tell the operators is "If the text is WHITE, don’t believe the reading. The computer cannot communicate to the process variable". Therefore, white static text is discouraged.

Likewise, since RED, GREEN, and YELLOW reflect alarm conditions, these colors should not be used for static text or text updates.

No colors are "disallowed" from being used for graphical images (magnets, waveguides, etc) because it is usually very obvious that these are pictures and NOT dynamic objects.

Colors in General

- DIN Standards 4844 and 5381 should be followed whenever possible.
- Red is a danger color. It means Halt, Stop, Prohibited, etc.
- Yellow is used as a warning color.
- Green means Safety, OK, On etc.
- Blue is used for giving directions, advice, signs, etc.

Environment Variables

MEDM uses the following environment variables:

EPICS_DISPLAY_PATH	A colon-separated (semi-colon-separated on WIN32) list of directories in which to look for display files that do not have absolute path names or names that begin with "./" or "../". Only looks in the current working directory if not specified. See ADL Files and Related Display.
MEDM_EXEC_LIST	A list of commands for the Execute-Mode Popup Menu. See the Execute Menu for the format.
MEDM_HELP	Name of the command used for the Help item in the Window Manager Menu Button for a display. This menu item does not appear if MEDM_HELP is not defined.
MEDM_MAIL_CMD	Name of the command used to send mail from dialog boxes that support it. The default is "mail".
NETSCAPEPATH	The name of the desired Netscape executable used for Help. It can be a full path name. The default is "netscape".
PSPRINTER	The default value for the printer in the Printer Setup dialog box. The default is "".

Building MEDM

MEDM is built with the EPICS build system. Explaining this system is beyond the scope of this manual. If you want to build MEDM, you should look at the EPICS documentation. The relevant manual is the current version of *IOC Applications: Building and Source/Release Control*. There are comments in some sections of this MEDM manual (Requirements, XRT/Graph, Sciplot, CDEV) on build parameters that are specific to MEDM.

MEDM has been successfully built on Solaris, Sun 4, HPUX, Linux, VMS, and WIN32.

CDEV Support

MEDM can be built to support CDEV rather than EPICS Channel Access. This capability has been provided by Jie Chen at the Thomas Jefferson National Accelerator Facility, and questions about the CDEV features of MEDM should be related to him at chen@jlab.org. CDEV provides an interface, not only to Channel Access, but also to any other network services that support CDEV at the expense of adding another layer to the control system. With CDEV support, you can specify messages in the form of "device verb attribute" where you would specify just a process variable name with Channel Access. In its simplest form, this message could be just the process variable name when interfacing to Channel Access. Apart from its supporting messages, the basic operation of MEDM is the same whether it is built for Channel Access or CDEV. ADL files that reference process variables by name should work with MEDM under CDEV. ADL files that use "device verb attribute" messages will not work with the non-CDEV MEDM.

In order to use MEDM with a CDEV service, the service has to support "get attribute" at least. If one wants to monitor a set of strings as choices, the service has to return a cdevData object with "displayHigh", "displayLow", and "value" tags in it. If there is not "displayHigh" and "displayLow", MEDM will not display data properly.

To build MEDM for CDEV, you need to define the makefile variable MEDM_CDEV. Setting it to YES is suggested.

Glossary

ADL Stands for ASCII Display List. The file format or the file that describes a display. The filename ends in .adl.

CDEV Stands for Control Device Interface. CDEV is a set of object-oriented software tools providing an interface to underlying systems, typically control system interfaces, such as Channel Access. CDEV has primarily been developed at the Thomas Jefferson National Accelerator Facility in collaboration with the EPICS community. The CDEV home page is http://www.jlab.org/cdev.

Channel Access A set of routines to provide network-independent access to process variables.

CDE Stands for the Common Desktop Environment. A technology specification developed jointly by Hewlett-Packard Company, IBM Corporation, Novell, Inc., and SunSoft, Inc. It defines a consistent set of application programming interfaces (API) that can be implemented on operating environments that support X Windows desktop computers and OSF Motif. Its intent is to provide users with a consistent graphical user interface across workstations, X-Terminals, and PC?s.

Display In MEDM a display is a window on the computer screen that contains objects that monitor or control process variables. This MEDM definition should be contrasted with the X Window?s definition of a display, which is a set of one or more screens and a keyboard driven by a single X server, or the X Window DISPLAY environmental variable, which represents a particular screen on the X display.

EPICS Stands for Experimental Physics and Industrial Control System. EPICS is a set of software tools and applications originally developed by Argonne National Laboratory and Los Alamos National Laboratory for the purpose of building distributed control systems to operate devices such as Particle Accelerators, Large Experiments, and Telescopes.

EPICS Extension One of a number of programs that are not part of EPICS base but which have been designed to work intimately with EPICS. MEDM is an example. Most of the extensions are described in the online EPICS documentation.

IOC Stands for Input/Output Controller. A VME/VXI based chassis containing a Motorola 68xxx processor, various I/O modules, and VME modules that provide access to other I/O busses. It contain a memory-resident database of control records.

Motif Motif is a complete set of widgets supplied by the Open Software Foundation (OSF) designed to implement the application look and feel specified in the *Motif Style Guide* and *Motif Application Environment Specification*. The routines are based on the Xt Intrinsics.

Process variable The name of a field of a record in a database, usually but not necessarily an IOC database. The name is of the form record.field. If the field is missing, it is assumed to be .VAL.

Object An MEDM object is one of several entities, such as a Meter, Related Display, Cartesian plot, etc., which can appear on an MEDM display.

WIN32 The Microsoft Windows 95, Windows 98, and Windows NT operating systems.

Widget An interface component, such as a scroll bar, menu, or button on a user interface. Most MEDM objects are widgets and are children of the drawing area of the display. The others are drawn on the drawing area of the display.

Xt Intrinsics A basic set of routines for building and using widgets.

X Windows A network-based, graphics windowing system for workstations. It was developed by MIT and has become an industry standard.

Technical Support

If you have problems, comments, or questions about MEDM you can send them to evans@aps.anl.gov. If you wish to report a bug, it is essential that you send enough information for the bug to be reproduced. It would be helpful to include the MEDM version number, how MEDM was started, the circumstances in which the problem occurs, and an explicit list of steps to duplicate the problem. A copy of the ADL file or how it can be found is usually helpful. The resources available for technical support are extremely limited. Please try hard to resolve problems yourself first and to prepare a complete and thought-out problem report otherwise.

Copyright

Modified September 21, 1999

Maintained by Ken Evans (evans@aps.anl.gov)