Visual ADE 1.0c

Software Farm, Inc

Examples

Hello World:

```
ApplicationDemo1
      {
      Window
            {
            Panel
                  {
                  Translation
                        {
                        create            AddPushButton(Hello World)
                        "Hello World"     Quit
                        }
                  }
            }
      }
```

Better Hello World:

```
MenuBar Demo2MenuBar
      {
      Definition
            {
            Files
                  {
                  Quit        quit
                  }
            Help
                  {
                  "On Help"   postMsgBox(This is a sample help message.);
                  }
            }
      }

ApplicationDemo2
      {
      Window Demo2Window
            {
            Demo2MenuBar
            Panel
                  {
                  Translation
                        {
                        create      Demo2Window::setTitle(Visual ADE- Demo2);
                        Create      AddPushButton(Hello World / Goodbye World)
                        "Hello World / Goodbye World"Quit
                        }
                  }
```

```
            }
      }
```

Another Demo:

```
Panel MyPanel
      {
      Panel
            {
            Translation
                  {
                  create      catchErrors(error);
                  create      setLayout(Current, Vertical, 3)
                  create      addLabel(Label)
                  create      addLabelIcon(terminal)
                  create      AddPushButton(Push Button);
                  create      addPushIcon(terminal)
                  create      AddToggleButton(Toggle Button);
                  create      AddToggleIcon(terminal);
                  create      AddOptionMenu(&MyOptionMenu)
                  }
            }
      Panel
            {
            Translation
                  {
                  create      addSeparator()
                  }
            }
      Panel
            {
            Translation
                  {
                  create      setLayout(Current, Vertical, 3)
                  create      AddPushButton(MsgBox);
                  MsgBox      postMsgBox(Sample MsgBox)
                  create      AddPushButton(TextEntry);
                  TextEntrypostTextEntryBox(Sample TextEntryBox)
                  create      AddPushButton(SelBox);
                  SelBox      BlockingExecInShell(ls *.dsc > tmp),
postSelectionBox(tmp);
                  }
            }
      }
Menu MyOptionMenu
      {
      Definition
            {
            Option1
                  {
                  Option1           PostMsgBox(The option "%1" is NOT
supported, InfoMsg);
                  ________
                  ________
```

```
                  Quit              quit
                  }
            Help
                  {
                  "On Help"   postMsgBox(There is no help here, InfoMsg);
                  "On Version"postMsgBox(Version 0.10, InfoMsg);
                  }
            }
      }
MenuBar MyMenuBar
      {
      Definition
            {
            Files
                  {
                  About...    about
                  ________
                  ________
                  Quit              quit
                  }
            Help
                  {
                  "On Help"   postMsgBox(This is a panel of just some of the
widgets available with VisualADE, InfoMsg);
                  "On Version"postMsgBox(Version 1.00, InfoMsg);
                  }
            }
      }
Window MyMainWindow
      {
      Translation
            {
            create            setTitle(Panel Widgets Demo);
            }
      MyPanel
      MyMenuBar
      }
ApplicationDemo3
      {
      MyMainWindow
      }
```

Demo #4

```
TreeGraph MyTree
      {
      Translation TreeXLate
        {
            Create      loadGraphicsFile(grdata1),
updateViewToIncludeAllGraphics();
            loadfilepostTextEntryBox("Enter name of graphics file to load:",
grdata, loadit);
```

```
            loadit      deleteContents(), loadGraphicsFile(%1),
updateViewToIncludeAllGraphics(), MyEditor::draw();
            savefilepostTextEntryBox("Enter name of file to save graphics
to:", grdata, saveit);
            saveit      saveGraphicsFile(%1);
            }
      }

Translation MyPanelXLate
      {
      create      AddPushButton(Pink);// comments here
      create      AddPushButton(Orange);
      create      AddToggleButton(Black & White);
      Pink        MYEDITOR::setbackgroundcolor(pink);
      Orange      MYEDITOR::setbackgroundcolor(orange);
      "Black & White"MYEDITOR::setbackgroundcolor(white);
      "~Black & White" MYEDITOR::setbackgroundcolor(black);
      create            AddToggleButton(2 More Views);
      "2 More Views"MyView1::hide(False), MyView2::hide(False);
      "~2 More Views"MyView1::hide(True), MyView2::hide(True);

      create            AddToggleButton(Locator & Scope);
      "~Locator & Scope"MyLocator::hide(True), MyScope::hide(True);
      "Locator & Scope"MyLocator::hide(False), MyScope::hide(False);

      create            AddPushButton(Zoom in Area);
      "Zoom in Area"MyEditor::centerCursor(), MyEditor::zoomThruArea(In,
start);

      create            AddPushButton(Import Rectangle);
      "Import Rectangle"MyTree::addGraphics(Writemode Replace)
      "Import Rectangle"MyTree::addGraphics(Rectangle 0 0 1000 1000)
      "Import Rectangle"MyEditor::centerCursor()
      "Import Rectangle"MyEditor::centerLast()
      "Import Rectangle"MyEditor::drawLast()
      "Import Rectangle"MyEditor::simpleDrag(start)

      create            AddPushButton(Import Graphics);
      "Import Graphics"MyTree::loadGraphicsFile(grdatafile)
      "Import Graphics"MyEditor::centerCursor()
      "Import Graphics"MyEditor::centerLast()
      "Import Graphics"MyEditor::drawLast()
      "Import Graphics"MyEditor::simpleDrag(start)

      doSomething printf("Timer\n");
      }
Panel MyPanel
      {
      MyPanelXLate
      }
Translation MyTranslation
      {
      <Defaults>setBackgroundColor(black);
      <Defaults>jumpPan
```

```
      <Defaults>selectArea();
      <Defaults>deselectAll();
      <Defaults>cumulativeSelect();
      <Defaults>zoomAroundCursor();
      <Defaults>zoomThruArea();
      <Defaults>treeNodeDrag();

      <Btn1Click>simpleDrag(end);
      <motion>simpleDrag(move);

      <Defaults>iCreateRect();

      <btn3Drag>MyPanel::sendTimerMsg(readtimer, Disable, readRouter2),
smoothPan();
      <btn3Up>MyPanel::sendTimerMsg(readtimer, Enable, readRouter2);
      <Motion>zoomThruArea(In, Move);
      <Btn1Click>zoomThruArea(In, End);
      Create      setBackgroundColor(gray);
      Create      setGraph(MyTree);
      ^<key>g     setBackgroundColor(gray);
      Ctrl<key>wsetBackgroundColor(white);
      <WinMap>hideTextIfTooSmall();
      Ctrl<key>mprintAllMessages();
      ^<key>d     dumpTranslations()
      <key>o      onePointPan()
      <key>r      MyTree::placeTopDown();
      <key>n      showNodes(Toggle), draw
      ^<key>s     MyTree::saveGraphicsFile(data2)
      Ctrl<key>nMyScope::locatorToolOptions(attachBoxToMouse, Toggle)
      Ctrl<key>aMyScope::locatorToolOptions(locatorHasBox, Toggle)
      Ctrl<key>eMyScope::locatorToolOptions(MaintainMagnification, Toggle)
      ^<key>g     MyLocator::locatorToolOptions(anotherGraphicsView, Toggle)
      ^<key>h     MyLocator::dumpTranslations()
      ^<key>i     MyScope::dumpTranslations()
      <Defaults>expandCursorFootprint();

      setbackcolorblockingExecInShell(cut -c14- /usr/lib/X11/rgb.txt | sed 's/
//g' > rgb), postSelectionBox(rgb, Select Color, setcolor)
      setcolorsetBackgroundColor(%1);
      }


Scope MyScope
      {
      Translation
            {
            Create      setSourceEditor(MyEditor);
            }
      }

Editor MyEditor
      {
      MyTranslation
      }
```

```
Locator MyLocator
      {
      Translation
            {
            Create      setSourceEditor(MyEditor);
            }
      }


View  MyView1
      {
      Translation
            {
            Create      setSourceEditor(MyEditor);
            }
      }
View  MyView2
      {
      Translation
            {
            Create      setSourceEditor(MyEditor);
            }
      }



Layout MyLayout
      {
      row(col(MyEditor, row(MyLocator, MyScope), row(MyView1, MyView2)),
MyPanel)

# Other possiblities...
#     col(MyEditor)
#     col(MyEditor, row(MyLocator, MyScope))
#     col(MyEditor, row(MyLocator, MyScope), row(View(MyEditor),
View(MyEditor)))
#     col(row(MyEditor, MyPanel), row(MyLocator, MyScope), row(View(MyEditor),
View(MyEditor)))
#     row(col(MyEditor, row(MyLocator, MyScope), row(View(MyEditor),
View(MyEditor))), MyPanel)
#     col(MyEditor, row(MyLocator, Scope(MyEditor)), row(View(MyEditor),
View(MyEditor)))
#     col(MyEditor, row(locator(MyEditor), scope(MyEditor)),
row(View(MyEditor), View(MyEditor)))
      }

MenuBar MyMenuBar
      {
      Definition
            {
            Files
                  {
                  Open                    MyTree::sendMsg(loadfile);
```

```
            Print...
            About...          about
            ________
            Save                    MyTree::saveGraphicsFile(data2);
            "Save As"         MyTree::sendMsg(savefile);
            !Close
            !New
            ________
            Quit                    quitWithChangesVerified
      }
      Edit
            {
            !Copy
            !Paste
            !Cut
            !Delete
            }
      Options
            {
            "Set Background Color..."MyEditor::sendMsg(setbackcolor);
            "Set Editor Size Small"MyEditor::setSize(300, 300)
            "Set Editor Size Medium"MyEditor::setSize(500, 500)
            "Set Editor Size Large"MyEditor::setSize(800, 800)
            }
      Help
            {
            "On Help"         postMsgBox(Click middle mouse button to
zoom in.\nClick right button to zoom out.\nMove mouse with right button down
to pan., InfoMsg);
            "On Version"      postMsgBox(Version 1.00, InfoMsg);
            }
      }
}

Translation MyWindowXlate
{
create      setTitle(Sample VisualADE Rectangle Editor);
}

Window MyMainWindow
{
MyWindowXlate
MyLayout
MyMenuBar
}
ApplicationDemo4
{
MyMainWindow
}
```

VisualADE 1.0c Reference Manual

Application

```
NAME:
     Application

SYNOPSIS:
     Application     [appname]

DESCRIPTION:
     Specifies the contents and name of the application that
     the description file describes.

FEATURES:
     The name (appname) of the application description can be
     referenced from the standard X Window System resource
     files.

MAY CONTAIN:
     Translation
     Window

CONTAINER COMPONENT:
     None.

CAVEATS:

SEE ALSO:
     Panel, Translation, Menubar, Window, Editor

PRIVATE MESSAGE HANDLERS:
     None.

FURTHER DESCRIPTION:

     The Application description must contain a window.

     For example:

             Application     Demo
                     {
                     MyWindow
                     }
```

Component

NAME:
Component

SYNOPSIS:
Components in Description Text Files

DESCRIPTION:
Description file Components correspond to large data and/or graphical objects that contain (sometimes lots of) smaller objects. A components description usually includes the description of its Translation which specifies actions (called Messages) and responses to actions (called Message Handlers) for the component.

FEATURES:
- Object Oriented

- Message Handlers (i.e. methods) can be added and removed from Components easily in the description file.

- Message Handlers are type checked against the component to assure that the Message Handler can do its job in the component you assign it to.

MAY CONTAIN:
Translation
(others are component type specific)

CONTAINER COMPONENT:
(depends on the type of component).

CAVEATS:

SEE ALSO:
Description, Application, Translation, Variable

EXPLANATION:

The description file consists of the description of some components. Components correspond to large data and/or graphical objects that contain (sometimes lots of) smaller objects. A components description usually includes the description of its Translation which specifies actions (called Messages) and responses to actions (called Message Handlers) for the component.

Components contain an arbitrarily large number of static data variables that can be used in the description file.

DblLinkList

NAME:
     DblLinkList

SYNOPSIS:
     DblLinkList   [listname]

DESCRIPTION:
     Specifies the contents and name of a graph that can be
     used in a Editor description.

FEATURES:

MAY CONTAIN:
     Translation

CONTAINER COMPONENT:
     Application

CAVEATS:

SEE ALSO:
     Translation, Editor, Application

PRIVATE MESSAGE HANDLERS:
     None.

FURTHER DESCRIPTION:

Definition

NAME:
     Definition

SYNOPSIS:
     Definition

DESCRIPTION:
     Defines the contents of it's parent component. For example
     when in a menubar description, this specifies the menubar
     contents and layout.

FEATURES:

MAY CONTAIN:

CONTAINER COMPONENT:
     Menubar

CAVEATS:

SEE ALSO:
     Window, Translation, Menubar

PRIVATE MESSAGE HANDLERS:
     None.

Description

NAME:
Description

SYNOPSIS:
Description Text Files

DESCRIPTION:
These files, in standard, editable ASCII, allow a person with a text editor to describe/modify/customize an application's appearance and behavior. Modifications may include changing what functionality is available in which editor, what events invoke the functionality, the number, type and layout of 'editing' areas, when and why dialog boxes are presented to the user, what to do when the user interacts with a dialog box, and more.

These modifications take effect as soon as the application is run again.

FEATURES:
Simplicity

MAY CONTAIN:
Application
Translation
Panel
Layout
MenuBar
Editor
Scope
Locator
Magnifier
View
Window
Menu
TreeGraph
DirectedGraph
UnDirectedGraph
DblLinkList

CONTAINER COMPONENT:
none.

CAVEATS:

SEE ALSO:
runVade, runVadeHandler, registerVadeHandler, Application
Translation, Component

EXPLANATION:

The description file consists of the description of some components. Components correspond to large data and/or graphical objects that contain (sometimes lots of) smaller objects. A components description usually includes the description of its Translation which specifies actions (called Messages) and responses to actions (called Message Handlers) for the component.

At the minimum, the description file must contain the description of an Application component.

In the description file:

#	In a line specifies that the rest of the line is a comment.
//	In a line specifies that the rest of the line is a comment.
{	Is the start of a description of a higher level object.
}	Is the end of a description of a higher level object.
aBc	All Messages and Message Handlers ignore the case of their names.
<empty lines>	Are Ignored.

DirectedGraph

NAME:
DirectedGraph

SYNOPSIS:
DirectedGraph [listname]

DESCRIPTION:
Specifies the contents and name of a graph that can be used in a Editor description.

FEATURES:

MAY CONTAIN:
Translation

CONTAINER COMPONENT:
Application

CAVEATS:

SEE ALSO:
Translation, Editor, Application

PRIVATE MESSAGE HANDLERS:
None.

FURTHER DESCRIPTION:

Editor

```
NAME:
     Editor

SYNOPSIS:
     Editor [editorname]

DESCRIPTION:
     Creates an area into which graphics objects can be placed
     and operated on.

FEATURES:

MAY CONTAIN:
     Translation

CONTAINER COMPONENT:
     Window
     Layout
     Application

CAVEATS:

SEE ALSO:
     Scope, Locator, View, Translation, Window,
     Layout, Application

PRIVATE MESSAGE HANDLERS:
```

Layout

```
NAME:
     Layout

SYNOPSIS:
     Layout [layoutname]

DESCRIPTION:
     Specifies the arrangement of the components contained within
     the layout definition.

     For example:

     row(component1, component2, col(component3, component4))

FEATURES:

MAY CONTAIN:
     Editor
     Locator
     Scope
     View
     Panel

CONTAINER COMPONENT:
     Window
     Panel

CAVEATS:

SEE ALSO:
     Window, Translation, Panel

PRIVATE MESSAGE HANDLERS:
```

Locator

NAME:
Locator

SYNOPSIS:
Locator [locatorname]

DESCRIPTION:
Creates an area which provides a birds-eye-view of the source editor graphics (see setSourceEditor Message Handler).

FEATURES:
Displays the relative position of the source editor viewport with respect to the entire graphics world by displaying a rectangle in the locator of the corresponding size.

Allows positioning and resizing of the source editor viewport by manipulation of the rectangle.

MAY CONTAIN:
Translation

CONTAINER COMPONENT:
Window
Layout
Application

CAVEATS:

SEE ALSO:
Editor, Scope, View, Translation, Window, Layout, Application

PRIVATE MESSAGE HANDLERS:

Menu

```
NAME:
     Menu

SYNOPSIS:
     Menu    [menuname]

DESCRIPTION:
     Specifies the contents and behavior (translations) a menu.

FEATURES:

MAY CONTAIN:
     Translation
     Definition

CONTAINER COMPONENT:
     None (referenced by addOptionMenu, addPopupMenu, ...).

CAVEATS:

SEE ALSO:
     Panel, Translation, Menubar

PRIVATE MESSAGE HANDLERS:

FURTHER DESCRIPTION:
     see Menubar.
```

Menu

```
NAME:
     Menu

SYNOPSIS:
     Menu    [menuname]

DESCRIPTION:
     Specifies the contents and behavior (translations) a menu.

FEATURES:

MAY CONTAIN:
     Translation
     Definition

CONTAINER COMPONENT:
     None (referenced by addOptionMenu, addPopupMenu, ...).

CAVEATS:

SEE ALSO:
     Panel, Translation, Menubar

PRIVATE MESSAGE HANDLERS:

FURTHER DESCRIPTION:
     see Menubar.
```

Menubar

```
   NAME:
        Menubar

   SYNOPSIS:
        Menubar [menubarname]

   DESCRIPTION:
        Creates a menubar and adds it to the containing window.

   FEATURES:

   MAY CONTAIN:
        Translation
        Definition

   CONTAINER COMPONENT:
        Window

   CAVEATS:

   SEE ALSO:
        Window, Translation, Definition

   PRIVATE MESSAGE HANDLERS:
        setSensitivity()

   FURTHER DESCRIPTION:

        Menubars are specified in the VisualADE description file in
        the following manner:

        Menubar [yourMenuBarName]
            {
            # Optional translations
            Translation
                {
                ...                             ...

                # Here a handler is specified to respond to the message
                # generated when the user selects a menubar item label
                Menu1Name.subMenu1Name.Label    MsgHandler
                ...                             ...

                # Menubar labels may be specified by their label name (if
                # unique) or their entire menu path name.
                externalEvent   setSensitivity(Label, False)
                externalEvent   setSensitivity(Menu1Name.subMenu1Name.Label2,
True)
                ...             ...
                }
```

```
aNameOfATranslation
...

# The specification of the menubar contents
Definition
    {
    # This name is displayed to the user as a
    # pulldown menu option in the menubar.
    Menu1Name
        {
        # Walking (cascading) menus may be specified.
        subMenu1Name
            {
            # Labels may specify their corresponding
            # translation here as a convenience.

            Label           MsgHandler

            ...

            # The exclamation point is used to initialize
            # the label to the desensitized state.

            !Label2         MsgHandler

            ...

            # Labels following the togglebutton keyword
            # denote toggle buttons.

            togglebutton

            ...

            # This 2nd label, following immediately after
            # another togglebutton label and having
            # the same name (label)
            # specifies the MsgHandler for the toggle
            # button 'unset' or 'out' position. If the
            # first label of the duo has a '~' (tilde)
            # then the button is initially not pressed,
            # otherwise it initially appears pressed.

            label3          MsgHandler
            ~label3         MsgHandler

            ...

            # Labels following the pushbutton keyword
            # denote push buttons.

            pushbutton
```

```
                ...

                # The solid line of all underscores specifies
                # that a separator line is to appear in the menu.

                ________

                ...
                }
            ...
            }
        # This name is also displayed as a pulldown
        # menu option in the menubar.
        Menu2Name
        ...
        }
    }
```

EXAMPLE:

```
Menubar MyMenubar
        {
        Definition
                {
                Files
                        {
                        Print
                                {
                                Postscript      printPS()
                                HPPCL           printPCL()
                                }
                        _____
                        _____
                        Quit                    quit()
                        }
                !Edit
                !View
                !Options
                !Help
                }
        }
```

Panel

```
NAME:
     Panel

SYNOPSIS:
     Panel [panelname]

DESCRIPTION:
     Creates an area into which user interface objects (widgets)
     can be placed.

MAY CONTAIN:
     Translation
     Panel
     Layout

CONTAINER COMPONENT:
     Panel
     Window

CAVEATS:

SEE ALSO:
     Translation, Window, Layout

PRIVATE MESSAGE HANDLERS:
```

Scope

```
NAME:
     Scope

SYNOPSIS:
     Scope [scopename]

DESCRIPTION:
     Creates an area which provides a magnified view of the
     source editor graphics (see setSourceEditor Message
     Handler).

FEATURES:

MAY CONTAIN:
     Translation

CONTAINER COMPONENT:
     Window
     Layout
     Application

CAVEATS:

SEE ALSO:
     Editor, Locator, View, Translation, Window,
     Layout, Application

PRIVATE MESSAGE HANDLERS:
```

Translation

NAME:
Translation

SYNOPSIS:
Translation [translationname]

DESCRIPTION:
Specifies the assignment of Message Handlers to Messages.

FEATURES:

MAY CONTAIN:
Editor
Locator
Scope
View
Panel

CONTAINER COMPONENT:
Window
Panel

CAVEATS:

SEE ALSO:
Window, Translation, Panel

EXPLANATION:

A Translation is a sequence of lines of the form:

```
  event | "msgname" | <defaults>    [Destination::]MsgHandler([parm1, ...]),
...
```

The left-hand-side specifies an event or message and the right-hand-side specifies what is to be done when the event occurs or the message is received.

The optional destination specifier must be a component which is capable of having a Translation. The specified MsgHandler must be valid for the destination (if one is specified) or valid for the containing component (if one is not). If it is not valid, a warning will be issued and the line will be ignored.

Multiple Message Handlers may be specified on one line if separated by commas. The Handlers are executed from left to right. Multiple Message Handlers may also be specified for a given Message by adding more lines, all with the same Message name on the left and the desired Message Handlers on the right.

Each Message causes the execution of from 0 to any number of an ordered list of Message Handlers. Each Message Handler, when executing, has the option of returning and not executing any of the following Message Handlers in the ordered list. Usually this only happens when an Exception is generated (see catchErrors).

Note that events must have a specifier within '<' and '>' (i.e. <key>a or <Btn1Drag>. Otherwise the 'event' will be interpreted as the name of a message.

An 'event' is generated by the user by using the keyboard or mouse.

A 'message' is generated either internally by the system or indirectly by the user (i.e. the menubar generates messages when a user selects an item).

'<Defaults>' indicates that the default translation, provided by the system, is to be used. The default translations provided are intended to be useful and coexist well so this option is often a quick way to add functionality. For example, specifying:

```
<Defaults>              jumpPan
```

automatically sets up to 16 different translations, which otherwise would have to be done manually by specifying each and every translation as in:

```
<LeftCursor>            jumpPan(quarterLeft)
<RightCursor>           jumpPan(quarterRight)
...                     ...
...                     ...
```

ASCII key events are supported as in:

```
<key>a
```

Event modifiers supported are

```
Shift
CapsLock
Ctrl
^
Alt
Meta
Btn1HeldDown
Btn2HeldDown
Btn3HeldDown
```

for example:

```
Ctrl<key>c      which is the same as ^<key>c
```

Special keys supported are:

```
        LeftArrow
        RightArrow
        UpArrow
        DownArrow
        Home
        End
        PgDn
        PgUp
        Delete
        Esc

    for example:

        <key>PgDn

Other events supported are:

        <Btn1Down>       <Btn2Down>        <Btn3Down>
        <Btn1Up>         <Btn2Up>          <Btn3Up>
        <Btn1Click>      <Btn2Click>       <Btn3Down>
        <Btn1Drag>       <Btn2Drag>        <Btn3Drag>
        <Btn1StartDrag>  <Btn2StartDrag>   <Btn3StartDrag>
        <Btn1DblClick>   <Btn2DblClick>    <Btn3DblClick>

        <Motion>
        <Repaint>
        <WinExit>
        <WinEnter>
        <WinResize>
        <WinMap>

    for example:

        <Motion>               cursorArm(Motion)
        quitmsg                Quit()
        <Btn2Click>            myEditor::zoom(In)

Special events:

        Create          The name of the message sent to
                        each component after it has been
                        created.
```

TreeGraph

NAME:
     TreeGraph

SYNOPSIS:
     TreeGraph   [treename]

DESCRIPTION:
     Specifies the contents and name of a graph that can be
     used in a Editor description.

FEATURES:

MAY CONTAIN:
     Translation

CONTAINER COMPONENT:
     Application

CAVEATS:

SEE ALSO:
     Translation, Editor, Application

PRIVATE MESSAGE HANDLERS:
     None.

FURTHER DESCRIPTION:

UnDirectedGraph

NAME:
     UnDirectedGraph

SYNOPSIS:
     UnDirectedGraph [listname]

DESCRIPTION:
     Specifies the contents and name of a graph that can be
     used in a Editor description.

FEATURES:

MAY CONTAIN:
     Translation

CONTAINER COMPONENT:
     Application

CAVEATS:

SEE ALSO:
     Translation, Editor, Application

PRIVATE MESSAGE HANDLERS:
     None.

FURTHER DESCRIPTION:

Variable

```
NAME:
     Variable

SYNOPSIS:
     Variables in Components in Description Text Files

DESCRIPTION:
     Description file Components contain a number of static
     variables that can be assigned to and read in the description
     file.

FEATURES:

MAY CONTAIN:

CONTAINER COMPONENT:
     Any.

CAVEATS:

SEE ALSO:
     Description, Application, Translation

EXPLANATION:

     These variables may be assigned values thus:

             setValue(%12, /usr/lib/X11/rgb.txt)

     this assigns the text "/usr/lib/X11/rgb.txt" to the
     variable %12.
     Variables may be accessed thus:

             exec(xterm -e vi %12)

     this 'dereferences' the %12 variable back into
     "/usr/lib/X11/rgb.txt" which is the passed to vi.
     At this time only numeric variable names are supported,
     all others are ignored.

     These variables, once assigned are kept around for the
     life of the executing program.

     Special Variables:

     %0      Is used to store the immediate history of the
             messages and message handlers (i.e. a call stack).

     %1      Is used by some of the Dialog boxes in FormADE to
             pass back a return value.
```

View

NAME:
View

SYNOPSIS:
View [viewname]

DESCRIPTION:
Creates an area which provides a another editable view of the source editor graphics (see setSourceEditor Message Handler).

FEATURES:

MAY CONTAIN:
Translation

CONTAINER COMPONENT:
Window
Layout
Application

CAVEATS:

SEE ALSO:
Editor, Locator, Scope, Translation, Window, Layout, Application

PRIVATE MESSAGE HANDLERS:

Window

NAME:
Window

SYNOPSIS:
Window [windowname]

DESCRIPTION:
Creates an window using the resident window system into which user interface objects (widgets) and graphics editors may be placed.

Window translations are sent a 'winClose' message when the closes (exits) the window by using the resident window manager system (i.e. selecting 'Close' from the menu in the window's border). This message may then be acted on like any other message. For example:

```
winClose        absorbMsg()
```

in a window's translation section causes the user's selection of the close option in this window to be ignored.

FEATURES:

MAY CONTAIN:
Translation
Panel
Layout
MenuBar
Editor

CONTAINER COMPONENT:
Application

CAVEATS:

SEE ALSO:
Editor, Translation, Layout, Application, MenuBar

PRIVATE MESSAGE HANDLERS:

centerCursor

```
NAME:
     centerCursor

SYNOPSIS:
     centerCursor()

DESCRIPTION:
     Sets the mouse cursor position to the center of the editor.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <key>a                  centerCursor()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     setCursor, setCursorAppearance, cursorArm, centerLast,
     centerLastUnderCursor
```

centerLast

```
NAME:
     centerLast

SYNOPSIS:
     centerLast()

DESCRIPTION:
     Sets the position of the last graphics primitive created
     to the center of the editor.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     centerLastUnderCursor, centerCursor, setCursor,
     setCursorAppearance, cursorArm
```

centerLastUnderCursor

```
NAME:
     centerLastUnderCursor

SYNOPSIS:
     centerLastUnderCursor()

DESCRIPTION:
     Sets the position of the last graphics primitive created
     to the position of the mouse cursor.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     centerLast, centerCursor, setCursor, setCursorAppearance,
     cursorArm
```

cumulativeSelect

```
NAME:
     cumulativeSelect

SYNOPSIS:
     cumulativeSelect()

DESCRIPTION:
     The user selects a group of objects by repeatedly selecting
     (by generating the event as specified in the translation)
     each object in turn. If a selected object is selected again,
     it is deselected.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Btn1Click>             cumulativeSelect()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     deselectAll, selectArea
```

cursorArm

```
NAME:
     cursorArm

SYNOPSIS:
     cursorArm(Motion | WinExit)

DESCRIPTION:
     'Arms' the graphics object underneath of the mouse cursor (if
     the graphics object is armable). This consists of a visual
     highlighting of the graphics object, indicating the fact that
     the mouse cursor is within the graphics object's selectable
     area.

PARAMETERS (Required):
     Motion                  The event that updates the currently
                             armed object, if it has changed.

     WinExit                 The event that assures there is no
                             currently armed object.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Motion>                cursorArm(Motion)
     <WinExit>               cursorArm(WinExit)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     setCursor, setCursorAppearance, centerCursor
```

deSelectAll

```
NAME:
     deSelectAll

SYNOPSIS:
     deSelectAll()

DESCRIPTION:
     Deselect (Un select) all objects currently selected.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Esc>                    deselectAll()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     cumulativeSelect, selectArea
```

draw

```
NAME:
     draw

SYNOPSIS:
     draw()

DESCRIPTION:
     Draw the graphics in the editing area, and all views, locator,
     scopes and magnifiers of the graphics.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <key>d                  draw()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     drawLast, showNodes, showOnlyRelatives
```

drawLast

NAME:
drawLast

SYNOPSIS:
drawLast()

DESCRIPTION:
Draw the last graphics primitive created.

PARAMETERS (Required):
None.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Editor

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:
draw, showNodes, showOnlyRelatives

expandCursorFootprint

NAME:
expandCursorFootprint

SYNOPSIS:
expandCursorFootprint()

DESCRIPTION:
Expands the apparent size of the cursor from a single pixel into a 2 by 2 pixel square. This allows the user's cursor work (picks) that are close to work as well as those that are exact.

PARAMETERS (Required):
None.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Editor

DEFAULT TRANSLATIONS:
<All> expandCursorFootprint().

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:

hide

```
NAME:
     hide

SYNOPSIS:
     hide(True | False)

DESCRIPTION:
     Hides or displays the entire editor.

PARAMETERS (Required):
     True                    The editor is hidden, the containing
                             window is resized to compensate.

     False                   The editor is displayed, the containing
                             window is resized to compensate.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     setSize, setBackgroundColor
```

hideTextIfTooSmall

```
NAME:
     hideTextIfTooSmall

SYNOPSIS:
     hideTextIfTooSmall()

DESCRIPTION:
     Tells all text graphics to draw only if bigger than or the
     same size as the text at the current zoom level. This is
     usually invoked when the window is first mapped. It allows
     the nodes in a graph, for example, to shrink when zoomed way
     out by not drawing the fixed-size-font text.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <WinMap>               hideTextIfTooSmall()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     draw
```

iCreateRect

NAME:
iCreateRect

SYNOPSIS:
iCreateRect(Start | Move | End)

DESCRIPTION:
Interactivly creates a rectangle by rubberbanding.

PARAMETERS (Required):

Start	The events that starts the interactive creation of the rectangle.
Move	The events that interactively changes the size of the rectangle.
End	The event that terminates the interactive creation of the rectangle.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Editor

DEFAULT TRANSLATIONS:

Shift<Btn1StartDrag>	iCreateRect(Start)
Shift<Btn1Drag>	iCreateRect(Move)
Shift<Btn1Up>	iCreateRect(End)

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:

jumpPan

```
NAME:
     jumpPan

SYNOPSIS:
     jumpPan( Left | Right | Up | Down | LeftSide | RightSide
             | Top | Bottom | LowerLeft | LowerRight | UpperLeft
             | UpperRight | QuarterLeft | QuarterRight | QuarterUp
             | QuarterDown)

DESCRIPTION:
     The editing area is panned by one of a number of fixed
     increments in one of a number of fixed directions.

PARAMETERS (Required):
     Left                    Pans one whole area left.
     Right                   Pans one whole area right.
     Up                      Pans one whole area up.
     Down                    Pans one whole area down.

     LeftSide                Pans all the way to the left edge.
     RightSide               Pans all the way to the right edge.
     Top                     Pans all the way to the top edge.
     Bottom                  Pans all the way to the bottom edge.

     LowerLeft               Pans one whole area to the lower left.
     LowerRight              Pans one whole area to the lower right.
     UpperLeft               Pans one whole area to the upper left.
     UpperRight              Pans one whole area to the upper right.

     QuarterLeft             Pans one quarter area left.
     QuarterRight            Pans one quarter area right.
     QuarterUp               Pans one quarter area up.
     QuarterDown             Pans one quarter area down.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
                             jumpPan(Left)
                             jumpPan(Right)
                             jumpPan(Up)
                             jumpPan(Down)

     Ctrl<LeftArrow>         jumpPan(LeftSide)
     Ctrl<RightArrow>        jumpPan(RightSide)
     Ctrl<UpArrow>           jumpPan(Top)
     Ctrl<DownArrow>         jumpPan(Bottom)
```

```
     <End>                    jumpPan(LowerLeft)
     <PgDn>                   jumpPan(LowerRight)
     <Home>                   jumpPan(UpperLeft)
     <PgUp>                   jumpPan(UpperRight)


     <LeftArrow>              QuarterLeft"
     <RightArrow>             QuarterRight"
     <UpArrow>                QuarterUp"
     <DownArrow>              QuarterDown"


MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     smoothPan, smoothPan2, onePointPan
```

locatorToolOptions

```
NAME:
     locatorToolOptions

SYNOPSIS:
     locatorToolOptions(optionname [, True | False | Toggle])

DESCRIPTION:
     Modifies the behavior of Locators and Scopes and Views.

PARAMETERS (Required):
     optionname      Name of an option. The following are
                     currently supported:

                             attachBoxToMouse
                             locatorHasBox
                             maintainMagnification
                             anotherGraphicsView

                     attachBoxToMouse:
                             Specifically for editors with a
                             Scope. Causes the scope to be
                             continuously updated to display
                             the graphics in the area of the
                             current mouse pointer position.

                     locatorHasBox:
                             Specifically for editors with a
                             Scope. Causes the scope to be
                             represented by a visible box
                             (rectangle) in the source editor.

                     maintainMagnification:
                             Specifically for editors with a
                             Scope. Causes the scope to be
                             automatically adjusted after zooms
                             in the source editor so that a
                             constant magnification factor is
                             applied.


                     anotherGraphicsView:
                             Specifically for editors with a
                             Locator. Causes the locator to
                             have a copy of the source editor
                             graphics as a background.


PARAMETERS (Optional):
     True            The option is applied to the editor(s).
```

False	The option is not applied to the editor(s).
Toggle	The option is toggled from True to False and back again each time Toggle is invoked. (when not specified is set to Toggle).

CONTAINER COMPONENT:

Editor
Locator
Scope
Magnifier
View

DEFAULT TRANSLATIONS:

None.

MESSAGES GENERATED:

None.

VARIABLES SET:

None.

EXCEPTIONS:

Class: PANEL,	Error: Missing Parameter
Class: PANEL,	Error: Bad Parameter.

CAVEATS:

SEE ALSO:

onePointPan

```
NAME:
     onePointPan

SYNOPSIS:
     onePointPan()

DESCRIPTION:
     The window is panned such that the graphics that were
     underneath the mouse cursor position before the pan is now
     in the center of the editing area.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <key>g                  onePointPan()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     smoothPan, smoothPan2, jumpPan
```

selectArea

```
NAME:
     selectArea

SYNOPSIS:
     selectArea(Start | Move | End)

DESCRIPTION:
     The user rubberbands a rectangle. All the graphics inside the
     rectangle is selected.

PARAMETERS (Required):
     Start                  The event that starts the interactive
                            creation of the rectangle.

     Move                   The events that interactively changes
                            the size of the rectangle.

     End                    The event that terminates the
                            interactive creation of the rectangle.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     Ctrl<Btn1StartDrag>    selectArea(Start)
     Ctrl<Btn1Drag>         selectArea(Move)
     Ctrl<Btn1Up>           selectArea(End)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     cumulativeSelect, deselectAll
```

setBackgroundColor

```
NAME:
     setBackgroundColor

SYNOPSIS:
     setBackgroundColor(color name)

DESCRIPTION:
     Sets the background color of the editing area to the color
     specified by the color name.

PARAMETERS (Required):
     color name              The name of the color to use.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     ^<key>b                 setBackgroundColor(black)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     setSize, hide
```

setSize

```
NAME:
     setSize

SYNOPSIS:
     setSize(width, height)

DESCRIPTION:
     Sets the size (width and height) of the editor in device
     coordinates.

PARAMETERS (Required):
     width                   The width of editor in pixels.

     height                  The height of editor in pixels.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: EDITOR,  Error: Missing Parameter

CAVEATS:

SEE ALSO:
     hide, setBackgroundColor
```

setSize

```
NAME:
     setSize

SYNOPSIS:
     setSize(width, height)

DESCRIPTION:
     Sets the size (width and height) of the editor in device
     coordinates.

PARAMETERS (Required):
     width                   The width of editor in pixels.

     height                  The height of editor in pixels.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: EDITOR,  Error: Missing Parameter
     Class: EDITOR,  Error: Bad Parameter

CAVEATS:

SEE ALSO:
     hide, setBackgroundColor
```

setSourceEditor

NAME:
setSourceEditor

SYNOPSIS:
setSourceEditor(name of editor)

DESCRIPTION:
Tells a Locator, Scope, View or Magnifier where the editor is whose graphics it is viewing. I.e. Specifies the editor which contains the graphics which is to have another 'view'.

PARAMETERS (Required):

name	The name of an Editor (as specified in the description file).

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Locator
Scope
Magnifier
View

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:

showNodes

NAME:
showNodes

SYNOPSIS:
showNodes(True, False, Toggle)

DESCRIPTION:
To display or not to display the nodes of a graph. If not displayed, then only the connections between the nodes will be visible.

PARAMETERS (Required):
None.

PARAMETERS (Optional):

True	The nodes are displayed.
False	The nodes are not displayed.
Toggle	If the nodes are not displayed then they are drawn, if they are displayed then they are hidden. (when not specified Toggle is chosen).

CONTAINER COMPONENT:
Editor

DEFAULT TRANSLATIONS:

<key>n	showNodes(Toggle)

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:
draw, showOnlyRelatives

showOnlyRelatives

```
NAME:
     showOnlyRelatives

SYNOPSIS:
     showOnlyRelatives(Children | Parents | All)

DESCRIPTION:
     For graphics composed of Graphs, this will hide all but the
     specified relatives of any selected graphic objects. Invoking
     this twice will display all the graphics again (i.e. this
     toggles the display automatically upon repeated invocation).

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     Children               The nodes that are not children of
                            the selected nodes are hidden.

     Parents                The nodes that are not parents of
                            the selected nodes are hidden.

     All                    The nodes that are not parents of
                            or children of the selected nodes
                            are hidden. T

                            (when not specified All is chosen).

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <key>k                 showOnlyRelatives(Children)
     <key>p                 showOnlyRelatives(Parents)
     <key>h                 showOnlyRelatives(All)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     draw, showNodes
```

simpleDrag

```
NAME:
     simpleDrag

SYNOPSIS:
     simpleDrag(Start | Move | End)

DESCRIPTION:
     Moves the graphics object underneath the mouse cursor
     interactively in response to the mouse. It assumes there are
     no connections to the graphics object and if there are, the
     connections are not automatically rubberbanded to maintain
     connectivity.

PARAMETERS (Required):
     Start                  The event that starts the move.

     Move                   The event that actual causes the
                            movement.

     End                    The event that terminates the drag
                            operation.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Btn1StartDrag>        simpleDrag(Start)
     <Btn1Drag>             simpleDrag(Move)
     <Btn1Up>               simpleDrag(End)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     treeNodeDrag
```

smoothPan

```
NAME:
     smoothPan

SYNOPSIS:
     smoothPan()

DESCRIPTION:
     The window is panned interactively in the direction opposite
     mouse cursor movement as if one was dragging around the
     viewport which was looking down on a section of the graphics.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Btn3Drag>              smoothPan()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     smoothPan2, onePointPan, jumpPan
```

smoothPan2

```
NAME:
     smoothPan2

SYNOPSIS:
     smoothPan2()

DESCRIPTION:
     The window is panned interactively in the direction opposite
     mouse cursor movement as if one was dragging around the
     viewport which was looking down on a area of the graphics.
     The area of the graphics that is displayed is proportional to
     the distance the mouse cursor is from the edge of the editor
     viewport (i.e. the scrollbar 'buttons' are continuously
     updated such that they center on the mouse cursor position).

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Btn3Drag>              smoothPan()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     smoothPan, onePointPan, jumpPan
```

treeNodeDrag

```
NAME:
     treeNodeDrag

SYNOPSIS:
     treeNodeDrag(Start | Move | End)

DESCRIPTION:
     Moves the graphics object underneath the mouse cursor
     interactively in response to the mouse. It takes account of
     the possibility that there are connections to the graphics object
     and 'rubberbands' the connections as the move occurs, keeping
     the object connected at all times.

PARAMETERS (Required):
     Start                  The event that starts the move.

     Move                   The event that actual causes the
                            movement.

     End                    The event that terminates the drag
                            operation.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Btn1StartDrag>        treeNodeDrag(Start)
     <Btn1Drag>             treeNodeDrag(Move)
     <Btn1Up>               treeNodeDrag(End)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     simpleDrag
```

zoom

```
NAME:
     zoom

SYNOPSIS:
     zoom(In | Out)

DESCRIPTION:
     Zoom (in: magnify, out: demagnify) graphics, centering the
     result around the graphics that was underneath the mouse
     cursor position before the zoom.

PARAMETERS (Required):
     In                    The event that causes magnification.

     Out                   The event that causes demagnification.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Editor

DEFAULT TRANSLATIONS:
     <Btn2Click>           zoom(In)
     <Btn3Click>           zoom(Out)

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     zoomAroundCursor, zoomThruArea
```

zoomAroundCursor

NAME:
zoomAroundCursor

SYNOPSIS:
zoomAroundCursor(In | Out)

DESCRIPTION:
Zoom, preserving the current mouse cursor position with respect to the underlying graphics. I.E. the graphics object(s) underneath the cursor before the zoom is underneath the cursor after the zoom.

PARAMETERS (Required):

In	The event that causes magnification.
Out	The event that causes demagnification.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Editor

DEFAULT TRANSLATIONS:

<Btn2Click>	zoom(In)
<Btn3Click>	zoom(Out)

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:
zoom, zoomThruArea

zoomThruArea

NAME:
zoomThruArea

SYNOPSIS:
zoomThruArea(In | Out, Start | Move | End)

DESCRIPTION:
The user rubberbands a rectangle. If it is a zoom in, then the window is zoomed such that the graphics in the rectangle fills the editing area. If it is a zoom out, the current window is zoomed such that the previous window contents now fill only the area of the rectangle.

PARAMETERS (Required):

In	The event that causes magnification.
Out	The event that causes demagnification.
Start	The events that starts the interactive creation of the rectangle that is to be used.
Move	The events that interactively changes the size of the rectangle.
End	The event that terminates the interactive creation of the rectangle that is to be used. This the triggers the actual zooming operation.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Editor

DEFAULT TRANSLATIONS:

<Btn2StartDrag>	zoomThruArea(In, Start)
<Btn2Drag>	zoomThruArea(In, Move)
<Btn2Up>	zoomThruArea(In, End)
<Btn3StartDrag>	zoomThruArea(Out, Start)
<Btn3Drag>	zoomThruArea(Out, Move)
<Btn3Up>	zoomThruArea(Out, End)

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

```
EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     zoomAroundCursor, zoomThruArea
```

addEditor

```
NAME:
     addEditor

SYNOPSIS:
     addEditor(&editorname)

DESCRIPTION:
     Creates the referenced editor and inserts it using the current layout
     template into the panel.

PARAMETERS (Required):
     editorname      Reference name of Editor.

PARAMETERS (Optional):

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter

CAVEATS:

SEE ALSO:
     Editor, Panel
```

addLabel

```
NAME:
     addLabel

SYNOPSIS:
     addLabel(name [, label [, value]])

DESCRIPTION:
     Creates a textual label and inserts it using the current layout
     template into the panel.

PARAMETERS (Required):
     name            Reference name of label.

PARAMETERS (Optional):
     label           Text to display.
                     (when not specified is set equal to the name
                     parameter).

     value           Text that is placed to the right of the label.
                     This can be modified by using 'setValue'.
                     (when not specified is set equal to nothing).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter

CAVEATS:

SEE ALSO:
     setLabel, addLabelIcon
```

addLabelIcon

```
NAME:
     addLabelIcon

SYNOPSIS:
     addLabelIcon(name[, filename])

DESCRIPTION:
     Creates a bitmap and inserts it using the current layout template
     into the panel.

PARAMETERS (Required):
     name            Reference name of Icon label.

PARAMETERS (Optional):
     filename        Path and filename where icon bitmap is defined.
                     (when not specified is set equal to the name
                     parameter).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: File Not Found.

CAVEATS:
     The bitmap pointed to by filename is in a format specified by
     the low level window system. For the X Window System, it is in
     the .xbm format.

SEE ALSO:
     addLabel, setLabel
```

addLayout

```
NAME:
     addLayout

SYNOPSIS:
     addLayout(name, container layout, Vertical | Horizontal | LabelValue,
             [, number of columns | number of rows[, justification]])

DESCRIPTION:
     Creates a imaginary template into which user interface objects
     (like push buttons and labels) can be inserted.

PARAMETERS (Required):
     name            Name of the layout.

     container       Name of the layout into which this is inserted.
                     Special names are:

                     Panel - The main panel layout.

                     Current - The current layout created by the last
                     'addLayout' (if there was one) or the Panel's
                     layout (if there was not).

PARAMETERS (Optional):
     orientation     How objects in this layout are to be laid out.
                     Can be set to:

                     Vertical - Objects are arranged into columns.
                     Horizontal - Objects are arranged into rows.
                     LabelValue - Objects are arranged into rows like:

                                  label : value
                                longlabel : value
                                   label2 : value button optionmenu

                     with the labels right justified in their column
                     and values left justified in theirs. Many objects,
                     such as AddLabel, AddTextField, have a value and an
                     optional value part which fit well into this scheme.
                     (when not specified is set equal to Vertical).

     number          Number of columns (if orientation is Vertical)
                     or number of rows (if orientation is Horizontal).
                     (when not specified is set equal to 1).

     justification   Alignment to be applied to items inserted into
                     the panel layout. Possible values are:

                             LeftJustified
                             RightJustified
```

CenterJustified
NotJustified

(when not specified is set equal to CenterJustified).

CONTAINER COMPONENT:
Panel

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
Class: PANEL, Error: Bad Parameter
Class: PANEL, Error: Missing Parameter

CAVEATS:

SEE ALSO:
setLayout, addPanel, addRadioLayout..

addOkCancelHelp

```
NAME:
     addOkCancelHelp

SYNOPSIS:
     addOkCancelHelp( 1 | 0 | okBtnName, 1 | 0 | applyBtnName, 1 | 0
             | cancelBtnName, 1 | 0 | helpBtnName [,helpMsgName] )

DESCRIPTION:
     Creates variations of the standard OK, Cancel, Apply, and Help
     push buttons in a row. Changes made to widgets in the enclosing
     window (including all panels) are not broadcast until a 'OK'
     or 'Apply' button is pressed. The 'Cancel' button causes the
     changes to be ignored and the window to be dismissed.

     If a BtnName is a '1' then the standard name is used for
     the button (i.e. 'OK'). If it is a '0' then the button is
     not displayed. Otherwise the name overrides the standard
     name with the specified name.

PARAMETERS (Required):
     okBtnName               '0' to disable, '1' for standard, name
                             to override standard.

     applyBtnName            '0' to disable, '1' for standard, name
                             to override standard.

     cancelBtnName           '0' to disable, '1' for standard, name
                             to override standard.

     helpBtnName             '0' to disable, '1' for standard, name
                             to override standard.

PARAMETERS (Optional):
     helpMsgName             The message to send when the helpBtn
                             is pressed by the user.
                             (when not specified is set equal to
                             'Help')

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     Help or helpMsgName     When the presses the 'Help' button.

VARIABLES SET:
     None.
```

```
EXCEPTIONS:
     Class: PANEL,           Error: Missing Parameter

CAVEATS:

SEE ALSO:
     addManualOkCancelHelp, addPushButton, sendMsg
```

addOptionMenu

```
NAME:
     addOptionMenu

SYNOPSIS:
     addOptionMenu(name, numoptions, options[, msgname [, label]])

DESCRIPTION:
     Creates an Option Menu and inserts it in the current layout
     structure into the panel.

PARAMETERS (Required):
     name            Reference name of Option Menu.

     numoptions      Number of names to put in the option menu.

     options         A list of names to put in the option menu.

PARAMETERS (Optional):
     msgname         Name of message to send when user selects an item
                     in the option menu.
                     (when not specified is set equal to nothing).

     label           Textual label to place to the left of this
                     option menu.
                     (when not specified is set equal to nothing).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         If a msgname was specified it is sent when the user
                     changes the option menu.

     <selection>     If no msgname was specified, the name of the user's
                     selection is sent as the message.

VARIABLES SET:
     %1              Set equal to the user's selection in the option menu
                     when a message is generated.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
```

addOptionMenu, addPopupMenu, addScrolledList, postSelectionBox, postMultiSelectionBox

addOptionMenu

```
NAME:
     addOptionMenu

SYNOPSIS:
     addOptionMenu(name, &menuname[, msgname [, label]])

DESCRIPTION:
     Creates an Option Menu and inserts it in the current layout
     structure into the panel.

PARAMETERS (Required):
     name            Reference name of Option Menu.

     menuname        Name of a MENU description.

PARAMETERS (Optional):
     msgname         Name of message to send when user selects an item
                     in the option menu.
                     (when not specified is set equal to nothing).

     label           Textual label to place to the left of this
                     option menu.
                     (when not specified is set equal to nothing).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         If a msgname was specified it is sent when the user
                     changes the option menu.

     <selection>     If no msgname was specified, the name of the user's
                     selection is sent as the message.

VARIABLES SET:
     %1              Set equal to the user's selection in the option menu
                     when a message is generated.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
     addOptionMenu, addPopupMenu, addScrolledList, postSelectionBox,
     postMultiSelectionBox
```

addPanel

```
NAME:
     addPanel

SYNOPSIS:
     addPanel(&panelName)

DESCRIPTION:
     Creates a panel from the referenced panel description and
     inserts it using the current layout template into the panel.

PARAMETERS (Required):
     name            Name of Panel description.

PARAMETERS (Optional):


CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter

CAVEATS:

SEE ALSO:
     addScrollPanel, addLayout, setLayout
```

addPopupMenu

```
NAME:
     addPopupMenu

SYNOPSIS:
     addPopupMenu(name, numoptions, options[, msgname])

DESCRIPTION:
     Creates an Popup Menu and inserts it in the current layout
     structure into the panel.

PARAMETERS (Required):
     name            Reference name of Popup Menu.

     numoptions      Number of names to put in the option menu.

     options         A list of names to put in the option menu.

PARAMETERS (Optional):
     msgname         Name of message to send when user selects an item
                     in the option menu.
                     (when not specified is set equal to nothing).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         If a msgname was specified it is sent when the user
                     changes the option menu.

     <selection>     If no msgname was specified, the name of the user's
                     selection is sent as the message.

VARIABLES SET:
     %1              Set equal to the user's selection in the option menu
                     when a message is generated.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
     addOptionMenu, addPopupMenu, addScrolledList, postSelectionBox,
     postMultiSelectionBox
```

addPopupMenu

```
NAME:
     addPopupMenu

SYNOPSIS:
     addPopupMenu(name, &menuname[, msgname])

DESCRIPTION:
     Creates an Popup Menu and inserts it in the current layout
     structure into the panel.

PARAMETERS (Required):
     name            Reference name of Popup Menu.

     menuname        Name of a MENU description.

PARAMETERS (Optional):
     msgname         Name of message to send when user selects an item
                     in the option menu.
                     (when not specified is set equal to nothing).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         If a msgname was specified it is sent when the user
                     changes the option menu.

     <selection>     If no msgname was specified, the name of the user's
                     selection is sent as the message.

VARIABLES SET:
     %1              Set equal to the user's selection in the option menu
                     when a message is generated.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
     addOptionMenu, addPopupMenu, addScrolledList, postSelectionBox,
     postMultiSelectionBox
```

addPushButton

```
NAME:
     addPushButton

SYNOPSIS:
     addPushButton(name [[, label], msgname])

DESCRIPTION:
     Creates a push button annotated with the given text label
     inserts it using the current layout template into the panel.

PARAMETERS (Required):
     name            Reference name of Button.

PARAMETERS (Optional):
     label           Path and filename where icon bitmap is defined.
                     (when not specified is set equal to the name
                     parameter).

     msgname         Name of message to send when this is selected
                     by user. (when not specified is set equal to
                     the name parameter).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         When user selects button.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter

CAVEATS:

SEE ALSO:
     addToggleButton, addPushIcon, addLabel, setLabel
```

addPushIcon

```
NAME:
     addPushIcon

SYNOPSIS:
     addPushIcon(name [[, filename], msgname])

PARAMETERS (Required):
     name            Reference name of Icon.

PARAMETERS (Optional):
     filename        Path and filename where icon bitmap is defined.
                     (when not specified is set equal to the name
                     parameter).

     msgname         Name of message to send when this is selected
                     by user. (when not specified is set equal to
                     the name parameter).

DESCRIPTION:
     Creates a push button annotated with a bitmap instead of a
     textual description and inserts it using the current layout template
     into the panel.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         When user selects button.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: File Not Found.

CAVEATS:
     The bitmap pointed to by filename is in a format specified by
     the low level window system. For the X Window System, it is in
     the .xbm format.

SEE ALSO:
     addPushButton, addToggleButton, addToggleIcon, addIcon, setLabel
```

addRadioLayout

```
   NAME:
        addRadioLayout

   SYNOPSIS:
        setRadioLayout(name, container layout, Vertical | Horizontal |
LabelValue,
                  [, number of columns | number of rows[, justification]])

   DESCRIPTION:
        Creates a imaginary template into which toggle buttons can
        be inserted. The toggle buttons exhibit radio button behavior
        so that one and only one button is selected at any time.

   PARAMETERS (Required):
        name            Name of the layout.

        container       Name of the layout into which this is inserted.
                        Special names are:

                        Panel - The main panel layout.

                        Current - The current layout created by the last
                        'addLayout' (if there was one) or the Panel's
                        layout (if there was not).

   PARAMETERS (Optional):
        orientation     How objects in this layout are to be laid out.
                        Can be set to:

                        Vertical - Objects are arranged into columns.
                        Horizontal - Objects are arranged into rows.

        number          Number of columns (if orientation is Vertical)
                        or number of rows (if orientation is Horizontal).
                        (when not specified is set equal to 1).

        justification   Alignment to be applied to items inserted into
                        the panel layout. Possible values are:

                                LeftJustified
                                RightJustified
                                CenterJustified
                                NotJustified

                        (when not specified is set equal to CenterJustified).

   CONTAINER COMPONENT:
        Panel

   DEFAULT TRANSLATIONS:
```

None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
Class: PANEL, Error: Bad Parameter
Class: PANEL, Error: Missing Parameter

CAVEATS:
Any other objects inserted into this layout cause an error message. The user is not prohibited from creating the radio layout with multiple toggle buttons 'set', however the behavior is then undefined.

SEE ALSO:
setLayout, addPanel.

addScrollPanel

```
NAME:
     addScrollPanel

SYNOPSIS:
     addScrollPanel(&panelName [, number columns [, number rows]])

DESCRIPTION:
     Creates a panel from the referenced panel description and
     inserts it using the current layout template into the panel.
     Vertical and Horizontal scrollbars appear when required
     (when the dimensions of the contents of the Scroll Panel
     exceed the dimensions of the Scroll Panel itself).

PARAMETERS (Required):
     name                    Name of Panel description.

PARAMETERS (Optional):
     number columns          The width of the scroll panel in characters
                             wide in the current text font.
                             (when not specified this is dynamically
                             calculated).

     number rows             The height of the scroll panel in characters
                             high in the current text font.
                             (when not specified this is equal to 1).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter

CAVEATS:

SEE ALSO:
     addPanel, addLayout, setLayout
```

addScrolledList

```
NAME:
     addScrolledList

SYNOPSIS:
     addScrolledList(filename, [number of rows [, msgname [, label]]])

DESCRIPTION:
     Creates an Scrolled List and inserts it in the current layout
     structure into the panel.

PARAMETERS (Required):
     filename        The path and name of a file where the list of
                     text to put in the scrolled list is to be
                     found.

PARAMETERS (Optional):
     numrows         The number of rows of text to display in the
                     list.
                     (when not specified is set equal to eight(8)).

     msgname         Name of message to send when user selects an item
                     in the option menu.
                     (when not specified is set equal to nothing).

     label           Textual label to place to the left of this
                     scrolled list.
                     (when not specified is set equal to nothing).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         If a msgname was specified it is sent when the user
                     changes the selection in the list.

     <selection>     If no msgname was specified, the name of the user's
                     selection is sent as the message.

VARIABLES SET:
     %1              Set equal to the user's selection in the list
                     when a message is generated.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter.
     Class: PANEL,   Error: File Not Found.

CAVEATS:
```

```
SEE ALSO:
     postSelectionBox, postMultiSelectionBox
```

addSeparator

NAME:
addSeparator

SYNOPSIS:
addSeparator()

DESCRIPTION:
Creates a horizontal line and inserts it using the current layout template into the panel.

PARAMETERS (Required):

PARAMETERS (Optional):

CONTAINER COMPONENT:
Panel

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:

CAVEATS:

SEE ALSO:
setLabel, addLabelIcon

addTextEditor

NAME:
addTextEditor

SYNOPSIS:
addTextEditor(name [, contents[, msgname[, label[, number columns[, number rows]]]]])

DESCRIPTION:
Creates a scrolled area for the user to enter a line of text and inserts it using the current layout template into the panel.

Note that for this Handler the following syntax is supported:

```
addTextEditor(name,,,,,5)
```

which specifies a text editor without contents, msgname is the same as the name, no label, number of columns dynamically determined and having 5 rows of text.

PARAMETERS (Required):

name	Reference name of text field.

PARAMETERS (Optional):

contents	Text to display in the text entry area. (when not specified is set equal to no text).
msgname	Name of message to send when this is exited by user. (when not specified is set equal to the name parameter).
label	Name of a label to place at left of text field. (when not specified is set equal to nothing).
num columns	Number of characters of text to display. (when not specified is dynamically determined).
num rows	Number of rows of text to display. (when not specified is set equal to 1).

CONTAINER COMPONENT:
Panel

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:

msgname	When user presses <Enter> or <Return>.

VARIABLES SET:

%1	Set equal to contents of text editor when message

is generated.

EXCEPTIONS:
Class: PANEL, Error: Missing Parameter

CAVEATS:
Messages should be sent when user exits text field.

SEE ALSO:
setLabel, addLabelIcon, addTextField

addTextField

```
   NAME:
        addTextField

   SYNOPSIS:
        addTextField(name [, contents[, msgname[, label[, number columns[,
number rows]]]]])

   DESCRIPTION:
        Creates an area for the user to enter a line of text and inserts
        it using the current layout template into the panel.

        Note that for this Handler the following syntax is supported:

                addTextField(name,,,,,5)

        which specifies a text field without contents, msgname is the same
        as the name, no label, number of columns dynamically determined
        and having 5 rows of text.

   PARAMETERS (Required):
        name            Reference name of text field.

   PARAMETERS (Optional):
        contents        Text to display in the text entry area.
                        (when not specified is set equal to no text).

        msgname         Name of message to send when this is exited
                        by user. (when not specified is set equal to
                        the name parameter).

        label           Name of a label to place at left of text field.
                        (when not specified is set equal to nothing).

        num columns     Number of characters of text to display.
                        (when not specified is dynamically determined).

        num rows        Number of rows of text to display.
                        (when not specified is set equal to 1).

   CONTAINER COMPONENT:
        Panel

   DEFAULT TRANSLATIONS:
        None.

   MESSAGES GENERATED:
        msgname         When user presses <Enter> or <Return>.

   VARIABLES SET:
        %1              Set equal to contents of text field when message
```

is generated.

EXCEPTIONS:
Class: PANEL, Error: Missing Parameter

CAVEATS:
Messages should be sent when user exits text field.

SEE ALSO:
setLabel, addLabelIcon, addTextEditor

addToggleIcon

```
NAME:
     addToggleIcon

SYNOPSIS:
     addToggleIcon(name [[, filename], msgname])

DESCRIPTION:
     Creates a toggle button annotated with a bitmap instead of a
     textual description and inserts it using the current layout template
     into the panel.

PARAMETERS (Required):
     name            Reference name of Icon.

PARAMETERS (Optional):
     filename        Path and filename where icon bitmap is defined.
                     (when not specified is set equal to the name
                     parameter).

     msgname         Name of message to send when this is selected
                     by user. (when not specified is set equal to
                     the name parameter).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         When user selects button.
     ~msgname        When user de-selects button.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: File Not Found.

CAVEATS:
     The bitmap pointed to by filename is in a format specified by
     the low level window system. For the X Window System, it is in
     the .xbm format.

SEE ALSO:
     addToggleButton, addPushButton, addToggleIcon, addIcon, setLabel
```

dismissWindow

```
NAME:
     dismissWindow

SYNOPSIS:
     dismissWindow(&windowname)

DESCRIPTION:
     Closes up the specified window. The window must have been
     previously posted.

PARAMETERS (Required):
     windowname              Name of a window or mainwindow as
                             defined in the description file.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter
     Class: GENERAL, Error: Bad State

CAVEATS:

SEE ALSO:
     postWindow.
```

getValue

```
NAME:
     getValue

SYNOPSIS:
     getValue(name)

DESCRIPTION:
     Gets the value of an object (widget) in the panel and assigns
     it to %1.

PARAMETERS (Required):
     name            Reference name of an Object in the Panel.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:
     None.

SEE ALSO:
     setLabel, setValue
```

manualAddOkCancelHelp

```
NAME:
     manualAddOkCancelHelp

SYNOPSIS:
     manualAddOkCancelHelp()

DESCRIPTION:
     Creates 3 push buttons in a row with the labels OK, Cancel, and
     Help, as per the OSF/Motif standard.

PARAMETERS (Required):

PARAMETERS (Optional):

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     Ok              When the presses the 'OK' button.
     Cancel          When the presses the 'Cancel' button.
     Help            When the presses the 'Help' button.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:
     This does NOT prohibit the sending of any message from user
     interface objects (widgets) in the panel before the 'OK'
     button acknowledgment is received from the user.

SEE ALSO:
     addOkCancelHelp, addPushButton, sendMsg
```

modifyFrame

```
NAME:
     modifyFrame

SYNOPSIS:
     modifyFrame(frameName, [frame width [, frame type]])

DESCRIPTION:
     Modifies the appearance of a Panel Layout's frame (visible
     rectangle surrounding the Panel).

PARAMETERS (Required):
     frameName       Name of the frame. Supported names are:

                             Current
                             Panel

PARAMETERS (Optional):
     width           The width, in pixels, of the frame.
                     (when not specified is set equal to zero(0)).

     type            The appearance of the frame. Can be one of:
                             ShadowIn
                             ShadowOut
                             ShadowEtchedIn
                             ShadowEtchedOut
                     (when not specified is set equal to ShadowIn).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Bad Parameter.
     Class: PANEL,   Error: Missing Parameter.

CAVEATS:

SEE ALSO:
     setColor, setBackgroundColor
```

postMsgBox

```
NAME:
     postMsgBox

SYNOPSIS:
     postMsgBox(text message[, message box type])

DESCRIPTION:
     Pops up (displays) a Message Dialog containing the specified
     text. The dialog displays an OK button and an Cancel button.

     When the user selects a button, the dialog disappears and if
     it is the Cancel button then this terminates the current
     thread of execution. In effect, this prohibits the execution
     of any handlers after this one from responding to the current
     message(the message that caused this Message Handler to execute).

PARAMETERS (Required):
     text            Message to display to user.

PARAMETERS (Optional):
     type            Kind of message box dialog to display. Possible
                     types are:

                             ErrorMsg
                             InfoMsg
                             JustAMsg
                             QueryMsg
                             WarningMsg
                             WorkingMsg

                     (when not specified is set equal to JustAMsg).

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter.

CAVEATS:

SEE ALSO:
```

```
postTextEntryBox
```

postMultiSelectionBox

```
   NAME:
        postMultiSelectionBox

   SYNOPSIS:
        postMultiSelectionBox(filename, [text prompt [, msgname]])

   DESCRIPTION:
        Pops up (displays) a Dialog Box containing a scrolled list
        of items found in the specified filename, the specified
        text prompt and an area for the user to respond. The user
        may use the mouse to select multiple items from the list.

        When the user selects a button, the dialog disappears and if
        it is the Cancel button then this terminates the current
        thread of execution. In effect, this prohibits the execution
        of any handlers after this one from responding to the current
        message(the message that caused this Message Handler to execute).

   PARAMETERS (Required):
        filename        The path and name of a file where the list of
                        text to put in the scrolled list is to be
                        found.

   PARAMETERS (Optional):
        text            Message to display to user in the window border title.
                        (when not specified is set equal to "Select An
Item:").

        msgname         Name of message to send when user OKs the dialog
                        box.
                        (when not specified is set equal to nothing).


   CONTAINER COMPONENT:
        Any

   DEFAULT TRANSLATIONS:
        None.

   MESSAGES GENERATED:

        (For each selected item in the list)

        msgname         If a msgname was specified it is sent when the user
                        changes the option menu.

        <selection>     If msgname was specified as '%1', the name of the
user's
                        selection is sent as the message.
```

```
     Help            When the user selects the Help button.

VARIABLES SET:
     %1              Set equal to the user's selection in the list
                     when a message is generated and when the user exits
                     the dialog box.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: File Not Found

CAVEATS:

SEE ALSO:
     postMsgBox, postTextEntryBox, postSelectionBox
```

postSelectionBox

```
   NAME:
        postSelectionBox

   SYNOPSIS:
        postSelectionBox(filename, [text prompt [, msgname]])

   DESCRIPTION:
        Pops up (displays) a Dialog Box containing a scrolled list
        of items found in the specified filename, the specified
        text prompt and an area for the user to respond. The user
        may use the mouse to select a single item from the list.

        When the user selects a button, the dialog disappears and if
        it is the Cancel button then this terminates the current
        thread of execution. In effect, this prohibits the execution
        of any handlers after this one from responding to the current
        message(the message that caused this Message Handler to execute).

   PARAMETERS (Required):
        filename        The path and name of a file where the list of
                        text to put in the scrolled list is to be
                        found.

   PARAMETERS (Optional):
        text            Message to display to user in the window border title.
                        (when not specified is set equal to "Select An
Item:").

        msgname         Name of message to send when user OKs the dialog
                        box.
                        (when not specified is set equal to nothing).


   CONTAINER COMPONENT:
        Any

   DEFAULT TRANSLATIONS:
        None.

   MESSAGES GENERATED:
        msgname         If a msgname was specified it is sent when the user
                        changes the option menu.

        <selection>     If msgname was specified as '%1', the name of the
user's
                        selection is sent as the message.

        Help            When the user selects the Help button.

   VARIABLES SET:
```

```
     %1              Set equal to the user's selection in the list
                     when a message is generated and when the user exits
                     the dialog box.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: File Not Found

CAVEATS:

SEE ALSO:
     postMsgBox, postTextEntryBox, postMultiSelectionBox
```

postTextEntryBox

```
   NAME:
        postTextEntryBox

   SYNOPSIS:
        postTextEntryBox([text prompt [, textfield contents [, msgname]])

   DESCRIPTION:
        Pops up (displays) a Dialog Box containing the specified
        text prompt and an area for the user to respond.

        When the user selects a button, the dialog disappears and if
        it is the Cancel button then this terminates the current
        thread of execution. In effect, this prohibits the execution
        of any handlers after this one from responding to the current
        message(the message that caused this Message Handler to execute).

   PARAMETERS (Required):
        None.

   PARAMETERS (Optional):
        text            Message to display to user.
                        (when not specified is set equal to "Enter Text:").

        contents        Default contents of the text entry field.
                        (when not specified is set equal to nothing).

        msgname         Name of message to send when user OKs the dialog
                        box.
                        (when not specified is set equal to nothing).


   CONTAINER COMPONENT:
        Any

   DEFAULT TRANSLATIONS:
        None.

   MESSAGES GENERATED:
        msgname         If a msgname was specified it is sent when the user
                        changes the option menu.

        <selection>     If msgname was specified as '%1', the name of the
user's
                        selection is sent as the message.

   VARIABLES SET:
        %1              Set equal to the user's textual response in the
                        textfield when a message is generated and when the
                        user exits the dialog box.
```

```
EXCEPTIONS:

CAVEATS:
     This is a blocking (modal) only Dialog.

SEE ALSO:
     postMsgBox, postSelectionBox, postMultiSelectionBox
```

postWindow

```
NAME:
     postWindow

SYNOPSIS:
     postWindow(&windowname)

DESCRIPTION:
     Pops up the specified window. If already displayed then pops
     the window to the front of any others currently displayed.

PARAMETERS (Required):
     windowname              Name of a window or mainwindow as
                             defined in the description file.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter

CAVEATS:

SEE ALSO:
     closeWindow.
```

setBackgroundColor

```
NAME:
     setBackgroundColor

SYNOPSIS:
     setBackgroundColor(name of an object, name of color)

DESCRIPTION:
     Assigns the background color of the named object (widget)
     in the Panel to the specified color. If no object is named
     then the background of the current layout is set to the given
     color.

PARAMETERS (Required):
     color           Name of a color.

PARAMETERS (Optional):
     name           Reference name of an Object in the Panel.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
     setColor, modifyFrame
```

setColor

```
NAME:
     setColor

SYNOPSIS:
     setColor(name of an object, name of color)

DESCRIPTION:
     Assigns the color of the named object (widget) in the
     Panel to the specified color. If no object is named
     then the current layout itself is set to the given color.

PARAMETERS (Required):
     color           Name of a color.

PARAMETERS (Optional):
     name            Reference name of an Object in the Panel.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
     setBackgroundColor, modifyFrame
```

setCursorAppearance

```
NAME:
     setCursorAppearance

SYNOPSIS:
     setCursorAppearance(cursor name)

DESCRIPTION:
     Sets the cursor appearance to be the specified image.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     cursor name     Name of cursor bitmap to replace the current one.
                     Possible names are system dependant. For example,
                     X Windows allows 'hand', 'watch', etc.
                     (when not specified is set equal to the previous
                     cursor image before the last change).


CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
```

setFont

```
   NAME:
        setFont

   SYNOPSIS:
        setFont([name of an object], [fontname])

   DESCRIPTION:
        Sets the font of the named object (widget) in the
        Panel to the specified font. If no object is named
        then all widgets in the current layout are set to
        have the given font.

   PARAMETERS (Required):

   PARAMETERS (Optional):
        name            Reference name of an Object in the Panel.
                         (when not specified is set to be all objects
                         subsequently created).

        fontname         Name of a font.
                         (when not specified is set equal to the default
                         system font).

   CONTAINER COMPONENT:
        Panel

   DEFAULT TRANSLATIONS:
        None.

   MESSAGES GENERATED:
        None.

   VARIABLES SET:
        None.

   EXCEPTIONS:
        Class: PANEL,   Error: Missing Parameter
        Class: PANEL,   Error: Bad Parameter.

   CAVEATS:
        Font names are system dependent. For X11/Motif systems a font
        name can be a system alias, an alias defined in a resource file
        (as the example below) or an actual font name (like the names
        that xlsfonts generates).

        For example a resource file for an application called Demo4
        might have:

        *Demo4*FontList:        -adobe-helvetica-medium-r-normal--14-140-75-
75-p-77-iso8859-1, \
```

```
                                -adobe-helvetica-bold-o-normal--14-140-75-75-
p-82-iso8859-1=demo4italic, \
                                -adobe-helvetica-bold-r-normal--14-140-75-75-
p-82-iso8859-1=demo4bold

        and the description file could then have:

                setFont(demo4bold)

        NOTE: specifying an actual font name does not work for
        certain versions of Motif widgets but does always work
        for text in VisualADE graphics editors.

   SEE ALSO:
        setColor, setBackgroundColor, modifyFrame
```

setLabel

```
NAME:
     setLabel

SYNOPSIS:
     setLabel(name, newlabel)

DESCRIPTION:
     Changes the label of an object (widget) in the panel.

PARAMETERS (Required):
     name            Reference name of an Object in the Panel.

     newlabel         Text of new label to assign to the named object.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:
     Does not work for iconic labels yet.

SEE ALSO:
     addLabel, setValue
```

setLayout

```
NAME:
     setLayout

SYNOPSIS:
     setLayout(name, Vertical | Horizontal | LabelValue,
             [, number of columns | number of rows[, justification]])

DESCRIPTION:
     Creates a imaginary template into which user interface objects
     (like push buttons and labels) can be inserted.

     Every Panel has a default layout. This default is the 'Current'
     layout until a 'addLayout' Message Handler is executed. At that
     time the new layout becomes the 'Current'.

PARAMETERS (Required):
     name            Name of the layout to modify. Special names are:

                     Current - The name of the current layout in effect
                     at the position of the 'setLayout' in the
                     translation.

PARAMETERS (Optional):
     orientation     How objects in this layout are to be laid out.
                     Can be set to:

                     Vertical - Objects are arranged into columns.
                     Horizontal - Objects are arranged into rows.
                     LabelValue - Objects are arranged into rows like:

                                  label : value
                                longlabel : value
                                   label2 : value button optionmenu

                     with the labels right justified in their column
                     and values left justified in theirs. Many objects,
                     such as AddLabel, AddTextField, optionally have a
                     value and an label part which fit well into this
                     scheme.

                     (when not specified is set equal to Vertical).

     number          Number of columns (if orientation is Vertical)
                     or number of rows (if orientation is Horizontal).
                     (when not specified is set equal to 1).

     justification   Alignment to be applied to items inserted into
                     the panel layout. Possible values are:

                             LeftJustified
```

RightJustified
CenterJustified
NotJustified

(when not specified is set equal to CenterJustified).

CONTAINER COMPONENT:
Panel

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
Class: PANEL, Error: Bad Parameter
Class: PANEL, Error: Missing Parameter

CAVEATS:

SEE ALSO:
addLayout, addPanel, addRadioLayout.

setMsgName

```
NAME:
     setMsgName

SYNOPSIS:
     setMsgName(name, new msgname)

DESCRIPTION:
     Changes the message that an object (widget) in the panel is to
     send when the user changes its state in some way.

PARAMETERS (Required):
     name            Reference name of an Object in the Panel.

     msgname          Name of new message to send when user interacts
                      with the object.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
     sendMsg, setLabel
```

setSensitivity

```
NAME:
     setSensitivity

SYNOPSIS:
     setSensitivity(name[, True | False])

DESCRIPTION:
     Permits or prohibits an object (widget) in the panel from
     interacting with the user. The object is usually 'grayed out'
     or in some way changes when interaction is prohibited.

PARAMETERS (Required):
     name            Reference name of an Object in the Panel.

PARAMETERS (Optional):
     True | False    True allows interaction with the object, False
                     disallows interaction with the object.
                     (when not specified is set equal to True).

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:

SEE ALSO:
```

setValue

```
NAME:
     setValue

SYNOPSIS:
     setValue(name, newvalue)

DESCRIPTION:
     Changes the value of an object (widget) in the panel.

PARAMETERS (Required):
     name            Reference name of an Object in the Panel.

     newvalue         Text of new value to assign to the named object.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Panel

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: PANEL,   Error: Missing Parameter
     Class: PANEL,   Error: Bad Parameter.

CAVEATS:
     None.

SEE ALSO:
     getValue, setLabel, addLabel
```

about

```
NAME:
     about

SYNOPSIS:
     about([text description])

DESCRIPTION:
     Posts an Information Dialog Box.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     text            The text to display in the dialog box.
                     (when not specified is set equal to a
                     message about VisualADE).

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     postMsgBox.
```

absorbMsg

NAME:
absorbMsg

SYNOPSIS:
absorbMsg()

DESCRIPTION:
Terminates the current thread of execution. In effect, this prohibits the execution of any handlers after this one from responding to the current message(the message that caused this Message Handler to execute).

PARAMETERS (Required):
None.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Any

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
Class: GENERAL, Error: Missing Parameter

CAVEATS:

SEE ALSO:
absorbMsg

blockingExec

```
NAME:
     blockingExec

SYNOPSIS:
     blockingExec(command line)

DESCRIPTION:
     Spawns a process and executes the specified command on the
     resident Operating System and waits for the finish of the
     execution before returning to VisualADE.

PARAMETERS (Required):
     command         Command to execute.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter
     Class: GENERAL, Error: System Error

CAVEATS:

SEE ALSO:
     exec, execInShell, blockingExecInShell
```

blockingExecInShell

```
NAME:
     blockingExecInShell

SYNOPSIS:
     blockingExecInShell(command line)

DESCRIPTION:
     Spawns a process and executes the specified command on the
     resident Operating System and waits for the finish of the
     execution before returning to VisualADE. The command is
     executed in a bourne ('sh') shell. This allows
     redirection and other shell features to be accessed
     directly from the handler command line.

PARAMETERS (Required):
     command         Command to execute.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter
     Class: GENERAL, Error: System Error

CAVEATS:
     Creates a temporary file called 'vadetmp' where the specified
     command line is stored. Then the command 'sh vadetmp' is
     executed where 'vadetmp' is then treated as a script file.
     This file is not automatically removed when this handler
     returns.

SEE ALSO:
     exec, execInShell, blockingExec
```

catchErrors

```
NAME:
     catchErrors

SYNOPSIS:
     catchErrors(msgname [[[, msgparm], msgparm]...])

DESCRIPTION:
     Sends the specified named message if this handler detects
     an exception by one of the Message Handlers following it
     in the flow of control. This is similar to the 'C++'
     language's exception handling scheme where exception handlers
     (like this catchErrors handler) get 'pushed and popped' on
     and off a stack of exception handlers as the control flow
     enters and exits the domains the exception handlers handle.

PARAMETERS (Required):
     msgname         Name of message to send.

PARAMETERS (Optional):
     msgparms        Name of parameters to the message to send.
                     (when not specified the message has no parameters).

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         As specified.

VARIABLES SET:
     %e0             General classname of type of component the
                     handler that generated the error is in.

     %e1             Name of the handler that generated the error.

     %e2             Text describing the creator of the handler.

     %e3             General description of the error.

     %e4             Severity of the error on a scale from 0 to 100.

     %e5             Detailed description of the error.

EXCEPTIONS:
     Class: GENERAL, Error: Not Initialized

CAVEATS:
     If an exception (i.e. error) is not 'caught' then a descriptive
```

```
     message about the exception is printed to standard out.

SEE ALSO:
     sendMsg, quitWithDeletesVerified, quitWithChangesVerified,
     absorbMsg, quit, Translation
```

dumpTranslations

```
NAME:
     dumpTranslations

SYNOPSIS:
     dumpTranslations()

DESCRIPTION:
     Prints to Standard Out in the window where VisualADE was
     started all translations that have been registered,
     internally and externally, with the containing component.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     All

DEFAULT TRANSLATIONS:
     ^<key>d                 dumpTranslations()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     printAllMessages
```

exec

```
NAME:
     exec

SYNOPSIS:
     exec(command line)

DESCRIPTION:
     Spawns a process and executes the specified command on the
     resident Operating System.

PARAMETERS (Required):
     command         Command to execute.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter
     Class: GENERAL, Error: System Error

CAVEATS:

SEE ALSO:
     execInShell, blockingExec, blockingExecInShell
```

execInShell

```
NAME:
     execInShell

SYNOPSIS:
     execInShell(command line)

DESCRIPTION:
     Spawns a process and executes the specified command on the
     resident Operating System in a bourne ('sh') shell. This
     allows redirection and other shell features to be accessed
     directly from the handler command line.

PARAMETERS (Required):
     command         Command to execute.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter
     Class: GENERAL, Error: System Error

CAVEATS:
     Creates a temporary file called 'vadetmp' where the specified
     command line is stored. Then the command 'sh vadetmp' is
     executed where 'vadetmp' is then treated as a script file.
     This file is not automatically removed when this handler
     returns.

SEE ALSO:
     exec, blockingExec, blockingExecInShell
```

printAllMessages

```
NAME:
     printAllMessages

SYNOPSIS:
     printAllMessages()

DESCRIPTION:
     Toggles the printing to Standard Out in the window where
     VisualADE was started all messages routed through the
     containing component.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     All

DEFAULT TRANSLATIONS:
     ^<key>m                 printAllMessages()

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     dumpTranslations
```

quit

```
NAME:
     quit

SYNOPSIS:
     quit()

DESCRIPTION:
     Terminates and exits the VisualADE program.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     quitWithChangesVerified, quitWithDeletesVerified.
```

quitWithChangesVerified

NAME:
quitWithChangesVerified

SYNOPSIS:
quitWithChangesVerified()

DESCRIPTION:
Checks to see if any of the Components in the system have changed, and if so, posts a Warning Dialog to the user to confirm or cancel exit. Otherwise the application is simply exited.

PARAMETERS (Required):
None.

PARAMETERS (Optional):
None.

CONTAINER COMPONENT:
Any

DEFAULT TRANSLATIONS:
None.

MESSAGES GENERATED:
None.

VARIABLES SET:
None.

EXCEPTIONS:
None.

CAVEATS:

SEE ALSO:
sendMsg, sendMsgIfChanges, quitWithDeletesVerified, absorbMsg, quit

quitWithDeletesVerified

```
NAME:
     quitWithDeletesVerified

SYNOPSIS:
     quitWithDeletesVerified()

DESCRIPTION:
     Sends the message named delete ("Delete") to all Components
     in the system and returns without continuing the current thread
     if any of the Components return a negative response. If a
     negative response is not returned the thread continues. It
     is intended that this be used as a supplement to
     quitWithChangesVerified by assigning a handler for the Delete
     message in Components that, for example, can inquire if
     application specific data has changed, posting a confirmation
     dialog to the user if so, and letting the dialog's cancel
     button return the negative response terminating the quit
     operation.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     Delete.         Sent to all components in the system.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     sendMsg, sendMsgIfChanges, quitWithChangesVerified,
     absorbMsg, quit
```

runCommand

```
NAME:
     runCommand

SYNOPSIS:
     runCommand(command)

DESCRIPTION:
     Executes the specified VisualADE command. This is useful for
     executing VisualADE Message Handlers from a dialog box.

PARAMETERS (Required):
     command         The command to execute. Allowable commands are
                     any Message Handler that is valid for the
                     containing component (i.e. anything that can
                     put on the right-hand-side of a translation).

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter

CAVEATS:

SEE ALSO:
     runCommandFile
```

runCommandFile

```
NAME:
     runCommandFile

SYNOPSIS:
     runCommandFile(filename)

DESCRIPTION:
     Executes the VisualADE commands found in the specified file.

PARAMETERS (Required):
     filename        The file containing the commands to execute.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: File Not Found

CAVEATS:

SEE ALSO:
     runCommand
```

sendMsg

```
NAME:
     sendMsg

SYNOPSIS:
     sendMsg(msgname)

DESCRIPTION:
     Sends the specified message to a component.

PARAMETERS (Required):
     msgname         The name of the message to send.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     As specified.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter

CAVEATS:

SEE ALSO:
     absorbMsg
```

sendMsgIfChanges

```
NAME:
     sendMsgIfChanges

SYNOPSIS:
     sendMsgIfChanges(msgname [[[, msgparm], msgparm]...])

DESCRIPTION:
     Sends the specified named message if this handler's container
     indicates that the user has made changes to it.

PARAMETERS (Required):
     msgname         Name of message to send.

PARAMETERS (Optional):
     msgparms        Name of parameters to the message to send.
                     (when not specified the message has no parameters).

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         As specified.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter

CAVEATS:

SEE ALSO:
     sendMsg, quitWithDeletesVerified, quitWithChangesVerified,
     absorbMsg, quit
```

sendTimerMsg

```
NAME:
     sendTimerMsg

SYNOPSIS:
     sendTimerMsg(name, time interval | Enable | Disable [, msgname])

DESCRIPTION:
     Sends the specified named message every interval of time
     if not disabled. This Message Handler can create a timer,
     disable a timer, or modify a timers interval and/or msgname.

PARAMETERS (Required):
     name            Name of the timer.

     time            Number of seconds between sending of the message.
                     This number is a real number (i.e. 0.1 causes
                     the message to be sent 10 times a second).
     Disable         The messages are not sent anymore.
     Enable          Cancels a previous Disable.

PARAMETERS (Optional):
     msgname         Name of the message to send.
                     (when not specified is set equal to the timer name
                     IF the timer has not been previously created,
                     otherwise is the same as previously set during
                     creation of the timer).

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     msgname         As specified.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter

CAVEATS:

SEE ALSO:
     sendMsg, quitWithDeletesVerified, quitWithChangesVerified,
     absorbMsg, quit
```

setSensitivity

```
NAME:
     setSensitivity

SYNOPSIS:
     setSensitivity(name[, True | False])

DESCRIPTION:
     Permits or prohibits an item (text) in the menubar from
     interacting with the user. The item is usually 'grayed out'
     or in some way changes when interaction is prohibited.

PARAMETERS (Required):
     name            Reference name of an item in the Menubar.
                      These names may be the same as the item text
                      or be specified using the cascaded path to
                      the item (i.e. Print.PostScript.Encapsulated)

PARAMETERS (Optional):
     True | False    True allows interaction with the item, False
                      disallows interaction with the item.
                      (when not specified is set equal to True).

CONTAINER COMPONENT:
     Menubar

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: MENUBAR, Error: Missing Parameter
     Class: MENUBAR, Error: Bad Parameter.

CAVEATS:

SEE ALSO:
```

setVariable

```
NAME:
     setVariable

SYNOPSIS:
     setVariable(destination variable name, source value)

DESCRIPTION:
     Assigns the specified value to the specified variable.

PARAMETERS (Required):
     name            The name of the variable. Allowable names are a
                     percent followed by any number:

                             %1, ..., %9, ... %12, ...

     value           The new value of the variable.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     As specified.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter

CAVEATS:

SEE ALSO:
     setVariableFromFile, saveVariableToFile
```

setVariableFromFile

```
NAME:
     setVariableFromFile

SYNOPSIS:
     setVariableFromFile(destination variable name, filename)

DESCRIPTION:
     Assigns the value from the specified file to the specified
     variable.

PARAMETERS (Required):
     name            The name of the variable. Allowable names are a
                     percent followed by any number:

                             %1, ..., %9, ... %12, ...

     filename        The filename where the value is found.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     Any

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     As specified.

EXCEPTIONS:
     Class: GENERAL, Error: Missing Parameter
     Class: GENERAL, Error: Bad Parameter

CAVEATS:

SEE ALSO:
     setVariable, saveVariableToFile
```

GraphicsFile

```
   NAME:
        GraphicsFile

   SYNOPSIS:
        The Textual Graphics File Format

   DESCRIPTION:
        A simple format for the storing of graphics data.

   CAVEATS:
        This is a flat file description at this time but will
        evolve into a hierarchical description in the future.

   SEE ALSO:
        loadGraphics, loadGraphicsFile, saveGraphicsFile

   EXPLANATION:

        The format of the file is one line per graphics primitive
        or attribute:

   <Primitive Type> [<Applications Name for Primitive>] <primitive data>

        for example:

                namedRectangle   couch    0 0 100 100

        the following variations in the format are also valid:

                namedRectangle,couch,0,0,100,100
                namedRectangle, couch, 0, 0, 100, 100

        1.      X and Y are coordinates which are integers -2147483648
                to +2147483647.
        2.      Color names are common colors like "red", "blue", etc.
        3.      Named graphics primitives do not display the application's
                name for the primitive, the name is for use as an 'ID'
                or 'key' in order to find particular primitives in the
                text graphics file and in memory.
        4.      Grid orientation:
                        1 = Horizontal steps (Vertical lines)
                        2 = Vertical steps (Horizontal lines)
                        3 = Horizontal and Vertical steps (crosshatch lines)

        the following graphics primitives and attributes are supported:

   AnnotatedRect, <Text Annotation, lowerleft X, lowerleft Y, upperright X,
upperright Y>

   namedAnnotatedRect, <Name of Rectangle, Text Annotation, lowerleft X,
lowerleft Y, upperright X, upperright Y>
```

Rectangle, <lowerleft X, lowerleft Y, upperright X, upperright Y>

namedRectangle, <Name of Rectangle, lowerleft X, lowerleft Y, upperright X, upperright Y>

Text, <Text String, Center X, Center Y>

namedText, <Name of Text, Text String, Center X, Center Y>

Line, <Start X, Start Y, End X, End Y>

namedLine, <Name of Line, Start X, Start Y, End X, End Y>

Circle, <Center X, Center Y, Radius>

namedCircle, <Name of Circle, Center X, Center Y, Radius>

Grid, <Orientation, X Step Size, Y Step Size, lowerleft X, lowerleft Y, upperright X, upperright Y>

namedGrid, <Name of Grid, Orientation, X Step Size, Y Step Size, lowerleft X, lowerleft Y, upperright X, upperright Y>

Color, <Name of Color>

FillColor, <Name of Color>

BackgroundColor, <Name of Color>

LineStyle, Solid | Dashed | DoubleDashed

LineWidth, <Width in Coordinates>

DeviceLineWidth, <Width in Pixels>

Filled, True | False

Writemode, Replace | Xor

addGraphics

```
NAME:
     addGraphics

SYNOPSIS:
     addGraphics(line describing graphics object or attribute)

DESCRIPTION:
     Creates and places the specified graphics object in the
     containing Graph or the specified attributes as the current
     'pen' color, linewidth, or whatever.

PARAMETERS (Required):
     description             A line of text based on the graphics
                             file format detailed in GraphicsFile.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GRAPH,   Error: Missing Parameter

CAVEATS:

SEE ALSO:
     loadGraphicsFile, saveGraphicsFile, GraphicsFile
```

assignGraphicsToGraph1

```
NAME:
     assignGraphicsToGraph1

SYNOPSIS:
     assignGraphicsToGraph1()

DESCRIPTION:
     Assigns graphics to the graph data using a built in procedure
     for assigning shadow rectangles annotated with text to nodes in
     the graph and straight lines to the connections (edges).

PARAMETERS (Required):
     None.

PARAMETERS (Optional):

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     deleteContents, loadGraphicsFile, saveGraphicsFile
```

deleteContents

```
NAME:
     deleteContents

SYNOPSIS:
     deleteContents()

DESCRIPTION:
     Deletes all graphics which has been assigned to graph data.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     assignGraphicsToGraph1, loadGraphicsFile, saveGraphicsFile.
```

loadDirGraph

```
NAME:
     loadDirGraph

SYNOPSIS:
     loadDirGraph(filename)

DESCRIPTION:
     Reads the specified file and creates and directed graph (or
     tree) from the file data.

PARAMETERS (Required):
     filename               Name of the file with the data in
                            the form:

                            child1  parent1 parent2
                            child2  parent1 parent3 parent4
                            child3  parent1 parent10

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GRAPH,   Error: Bad Parameter

CAVEATS:

SEE ALSO:
     loadDirGraph1, assignGraphicsToGraph1
```

loadDirGraph1

```
NAME:
     loadDirGraph1

SYNOPSIS:
     loadDirGraph1(filename)

DESCRIPTION:
     Reads the specified file and creates and directed graph (or
     tree) from the file data.

PARAMETERS (Required):
     filename        Name of the file with the data in
                     the form (similar to the 'du' utility):

              attribute1      grandparent1/parent1/child1
              attribute2      greatgrandparent2/grandparent2/parent2/child2
              attribute3      parent3/child3
              ...             ...

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GRAPH,   Error: Bad Parameter

CAVEATS:

SEE ALSO:
     loadDirGraph, assignGraphicsToGraph1
```

loadGraphicsFile

```
NAME:
     loadGraphicsFile

SYNOPSIS:
     loadGraphics(filename)

DESCRIPTION:
     Creates and places the graphics primitives as read from the
     specified filename and appends them to the containing
     Graph.

PARAMETERS (Required):
     filename                Name of the file with the data in
                             the file format detailed in
                             GraphicsFile.
                             (when not specified is set equal to
                             'grdata').

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GRAPH,   Error: File Not Found

CAVEATS:

SEE ALSO:
     loadGraphics, saveGraphicsFile, GraphicsFile
```

placeLeftToRight

```
NAME:
     placeLeftToRight

SYNOPSIS:
     placeLeftToRight()

DESCRIPTION:
     Places the contents of a directed graph or tree in a pleasant
     layout with the parents on the left and the children towards
     the right.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     placeTopDown
```

placeTopDown

```
NAME:
     placeTopDown

SYNOPSIS:
     placeTopDown()

DESCRIPTION:
     Places the contents of a directed graph or tree in a pleasant
     layout with the parents on the top and the children towards
     the bottom.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     placeLeftToRight
```

saveGraphicsFile

```
NAME:
     saveGraphicsFile

SYNOPSIS:
     saveGraphics(filename)

DESCRIPTION:
     Creates a textual representation of the graphics primitives
     found in the containing Graph writes it to the specified
     file. If the file already exists, the user is asked to
     confirm or cancel overwrite by a dialog box.

PARAMETERS (Required):
     filename               Name of the file to store the data
                            in the file format detailed in
                            GraphicsFile.
                            (when not specified is set equal to
                            'grdata').

PARAMETERS (Optional):
     None.

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     Class: GRAPH,   Error: System Error

CAVEATS:

SEE ALSO:
     loadGraphics, loadGraphicsFile, GraphicsFile
```

updateViewToIncludeAllGraphics

```
NAME:
     updateViewToIncludeAllGraphics

SYNOPSIS:
     updateViewToIncludeAllGraphics()

DESCRIPTION:
     Updates editor universe to assure inclusion all graphics which
     has been assigned to graph data.

PARAMETERS (Required):
     None.

PARAMETERS (Optional):

CONTAINER COMPONENT:
     UnDirectedGraph
     DirectedGraph
     TreeGraph
     DblLinkList

DEFAULT TRANSLATIONS:
     None.

MESSAGES GENERATED:
     None.

VARIABLES SET:
     None.

EXCEPTIONS:
     None.

CAVEATS:

SEE ALSO:
     assignGraphicsToGraph1, loadGraphicsFile, saveGraphicsFile.
```

VisualADE

```
   NAME:
        VisualADE - The Visual Application Development Engine

   SYNOPSIS:
        ade descriptionFilename

   DESCRIPTION:
        The VisualADE Engine is driven by an elegantly simple
        textual description file, that you use to describe the way
        you want your application to look and behave.

   FEATURES:

   SEE ALSO:
        Description, Application, Translation

   EXPLANATION:


compile
   NAME:
        compile

   SYNOPSIS:
        compile descriptionFile [outputFile] [-encrypt] [-header "your text"]

   DESCRIPTION:
        Converts the given description file into the format used by the
        runtime VisualADE engine. The output file generated by this
        utility can be distributed to users who do not have the
        development VisualADE. The output file cannot be modified by
        the users (and, optionally, cannot be examined either).

   INPUTS:
        descriptionFilename     The name of a description file.

        outputFilename          The name of the file to place the
                                compiled output. (If not specified
                                this is set to the descriptionFilename
                                with a '.ade' extension).

        -header                 Includes the text following this option
                                into the first line of the output file.

        -encrypt                Causes the output compiled file to be
                                encrypted (unreadable except for the
                                optional header).

   OUTPUTS:
        A description file in the format suitable for use with the
```

runtime version of VisualADE.

CAVEATS:

SEE ALSO:

registerVadeHandler

```
   NAME:
        registerVadeHandler

   SYNOPSIS:
        registerVadeHandler(EXTERNALFNPTR fn, char *name, char *client_data)

   DESCRIPTION:
        Available through the VisualADE library interface.

        Registers an application routine with the Visual ADE Engine. This
        routine can then be referenced by name in a description file. The
        routine is called whenever it is sent a message  by VisualADE's
        Engine (the parameter passing is described below).

   INPUTS:
        EXTERNALFNPTR   fn              The address of the function.
        char *          name            The name that can be used in the
                                        description file.
        char *          client_data     The pointer value to pass as a
parameter
                                        to the external function when it is
called.

   OUTPUTS:
        None.

   CALLBACK FUNCTION:

        <name>(char *client_data, int argc, char **argv)

        char *  client_data     As specified when the external function is
                                registered.
        int     argc            Number of arguments contained in argv array.
        char ** argv            Array of command line arguments. argv[0]
                                is the name of the function as registered.

        registerVadeHandler(name [[, label], msgname])

   CAVEATS:

   SEE ALSO:
        runVade, runVadeHandler

   EXAMPLE:

        {
        ...
        registerVadeHandler((EXTERNALFNPTR )&sample_external_fn, "printf",
NULL);
        runVade(argc, argv);
```

```
     }

Boolean sample_external_fn(char *client_data, int argc, char **argv)
     {
     client_data = client_data;

     printf(argv[1], argv[2], argv[3], argv[4], argv[5],
             NULL, NULL, NULL, NULL);

     return(True);
     }
```

runVade

```
NAME:
     runVade

SYNOPSIS:
     runVade(int argc, char **argv)

DESCRIPTION:
     Starts the Visual ADE Engine and does not return.

INPUTS:
     int     argc            Number of arguments contained in argv array.
     char ** argv            Array of command line arguments.

OUTPUTS:
     None.

CAVEATS:

SEE ALSO:
     registerVadeHandler, runVadeHandler
```

runVade

```
NAME:
     runVade

SYNOPSIS:
     runVade(int argc, char **argv)

DESCRIPTION:
     Starts the Visual ADE Engine and does not return.

INPUTS:
     int     argc            Number of arguments contained in argv array.
     char ** argv            Array of command line arguments.

OUTPUTS:
     None.

CAVEATS:

SEE ALSO:
     registerVadeHandler, runVadeHandler
```

runVadeHandler

```
NAME:
     runVadeHandler

SYNOPSIS:
     runVadeHandler(char *string)

DESCRIPTION:
     Available through the VisualADE library interface.

     Executes a Visual ADE Message Handler with the given name and
     parameters. Any text that would be valid on the right-hand-side
     of a translation can be used here.

INPUTS:
     char *  string          The Message Handler and parameters to
                             execute.

OUTPUTS:
     None.

CAVEATS:
     The Component assigned the executed Handler is called
     ExternalInterface and so a destination component must be provided
     explicitly for all but the general Message Handlers (i.e.
     MyPanel::setValue ...).

SEE ALSO:
     runVade, registerVadeHandler

EXAMPLE:
     runVadeHandler("MyPanel::setValue(%1, myFileName)")
```