## Contents

1 XAD Applications ............................................. 1
2 Mosel integration ........................................... 2
3 GUI creation with XAD ....................................... 4
4 Lifetime of XAD objects ..................................... 6
5 Events .......................................................... 7
6 XAD objects reference .........................................
   6.1 Browser .................................................................. 8
   6.2 Button ................................................................... 9
   6.3 Canvas .................................................................. 9
   6.4 Check button .......................................................... 10
   6.5 Drop list ............................................................... 10
   6.6 Editor .................................................................. 10
   6.7 Group ................................................................... 11
   6.8 Input .................................................................. 11
   6.9 List .................................................................... 12
   6.10 List with multiple columns ..................................... 12
   6.11 Progress bar ........................................................ 13
   6.12 Radio button ......................................................... 13
   6.13 Scroll bars ............................................................ 13
   6.14 Tab selectors ........................................................ 14
   6.15 Text .................................................................. 14
   6.16 Tree .................................................................. 15
   6.17 Window ............................................................... 15
   6.18 Subroutines specific to objects ......................... 16
      XADcreatetext ........................................................... 19
      XADtextaddtext ........................................................ 20
      XADtextgettext ......................................................... 21
      XADtextsettext ........................................................ 22
      XADcreatebutton ....................................................... 23
      XADcreateinput ........................................................ 24
      XADinputsettext ........................................................ 25
      XADinputgettext ....................................................... 26
      XADcreateeditor ........................................................ 27
      XADeditoraddtext ..................................................... 28
      XADeditorgettext ...................................................... 29
      XADcreateeditor ....................................................... 30
      XADcreateinput ........................................................ 31
      XADinputsettext ........................................................ 32
      XADinputgettext ....................................................... 33
      XADcreateeditor ........................................................ 34
      XADcreateeditor ........................................................ 35
      XADcreateeditor ........................................................ 36
      XADcreateeditor ........................................................ 37
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XADeditorsave</td>
<td>38</td>
</tr>
<tr>
<td>XADeditorsettext</td>
<td>39</td>
</tr>
<tr>
<td>XADeditorgettext</td>
<td>40</td>
</tr>
<tr>
<td>XADcreatecheck</td>
<td>41</td>
</tr>
<tr>
<td>XADchecksetstate</td>
<td>42</td>
</tr>
<tr>
<td>XADcheckgetstate</td>
<td>43</td>
</tr>
<tr>
<td>XADcreateradio</td>
<td>44</td>
</tr>
<tr>
<td>XADRadiosetstate</td>
<td>45</td>
</tr>
<tr>
<td>XADRadiogetstate</td>
<td>46</td>
</tr>
<tr>
<td>XADcreategroup</td>
<td>47</td>
</tr>
<tr>
<td>XADcreatelisting</td>
<td>48</td>
</tr>
<tr>
<td>XADlistadd</td>
<td>49</td>
</tr>
<tr>
<td>XADlistgetsel</td>
<td>50</td>
</tr>
<tr>
<td>XADlistselect</td>
<td>51</td>
</tr>
<tr>
<td>XADlistshow</td>
<td>52</td>
</tr>
<tr>
<td>XADcreatedroplist</td>
<td>53</td>
</tr>
<tr>
<td>XADdroplistadd</td>
<td>54</td>
</tr>
<tr>
<td>XADdroplistgetsel</td>
<td>55</td>
</tr>
<tr>
<td>XADdroplistselect</td>
<td>56</td>
</tr>
<tr>
<td>XADdroplistshow</td>
<td>57</td>
</tr>
<tr>
<td>XADcreateprogress</td>
<td>58</td>
</tr>
<tr>
<td>XADprogressset</td>
<td>59</td>
</tr>
<tr>
<td>XADcreatetab</td>
<td>60</td>
</tr>
<tr>
<td>XADtabgettab</td>
<td>61</td>
</tr>
<tr>
<td>XADtabsettab</td>
<td>62</td>
</tr>
<tr>
<td>XADcreatemultiline</td>
<td>63</td>
</tr>
<tr>
<td>XADmultilistshow</td>
<td>64</td>
</tr>
<tr>
<td>XADmultilistsetsize</td>
<td>65</td>
</tr>
<tr>
<td>XADmultilistsetcolname</td>
<td>66</td>
</tr>
<tr>
<td>XADmultilistsettext</td>
<td>67</td>
</tr>
<tr>
<td>XADmultilistrefresh</td>
<td>68</td>
</tr>
<tr>
<td>XADcreatecanvas</td>
<td>69</td>
</tr>
<tr>
<td>XADcanvasdrawbox</td>
<td>70</td>
</tr>
<tr>
<td>XADcanvasdrawellipse</td>
<td>71</td>
</tr>
<tr>
<td>XADcanvaserase</td>
<td>72</td>
</tr>
<tr>
<td>XADcanvasrefresh</td>
<td>73</td>
</tr>
<tr>
<td>XADcanvasdrawimage</td>
<td>74</td>
</tr>
<tr>
<td>XADcanvassaveimage</td>
<td>75</td>
</tr>
<tr>
<td>XADcanvasdrawline</td>
<td>76</td>
</tr>
<tr>
<td>XADcanvasdrawpoint</td>
<td>77</td>
</tr>
<tr>
<td>XADcanvasdrawrectangle</td>
<td>78</td>
</tr>
<tr>
<td>XADcanvasdrawpolygon</td>
<td>79</td>
</tr>
<tr>
<td>XADcanvasdrawpie</td>
<td>80</td>
</tr>
<tr>
<td>XADcanvasdrawarc</td>
<td>81</td>
</tr>
<tr>
<td>XADcanvasdrawchord</td>
<td>82</td>
</tr>
<tr>
<td>XADcanvasdrawtext</td>
<td>83</td>
</tr>
<tr>
<td>XADcanvasmap</td>
<td>84</td>
</tr>
<tr>
<td>XADcanvasunmap</td>
<td>85</td>
</tr>
<tr>
<td>XADcolor</td>
<td>86</td>
</tr>
<tr>
<td>XADcreatebrowser</td>
<td>87</td>
</tr>
<tr>
<td>XADbrowsergoto</td>
<td>88</td>
</tr>
<tr>
<td>XADcreatescrollbar</td>
<td>89</td>
</tr>
<tr>
<td>XADscrollbargetpos</td>
<td>90</td>
</tr>
<tr>
<td>XADscrollbarset</td>
<td>91</td>
</tr>
<tr>
<td>XADcreategroup</td>
<td>92</td>
</tr>
<tr>
<td>XADtreedad</td>
<td>93</td>
</tr>
<tr>
<td>XADtreereset</td>
<td>94</td>
</tr>
<tr>
<td>XADtreeexpand</td>
<td>95</td>
</tr>
</tbody>
</table>
6.19 Events specific to objects .................................................. 96
  XAD_EVENT_MENU .................................................. 96
  XAD_EVENT_TIMER .................................................. 96
  XAD_EVENT_WINDOW_CLOSED ........................................... 96
  XAD_EVENT_WINDOW_CLOSING .......................................... 96
  XAD_EVENT_WINDOW_HIDDEN ........................................... 97
  XAD_EVENT_WINDOW_MOVED ............................................ 97
  XAD_EVENT_WINDOW_OPENED ........................................... 97
  XAD_EVENT_WINDOW_RESIZED ........................................... 97
  XAD_EVENT_WINDOW_SHOWN ............................................ 97
  XAD_EVENT_PRESSED .................................................. 97
  XAD_EVENT_CHANGED .................................................. 98
  XAD_EVENT_SELECTION ................................................ 98

7 Generic routines .............................................................. 99
  XADdestroy ............................................................... 100
  XADenable ............................................................... 101
  XADgetmousex ........................................................... 102
  XADgetmousey ........................................................... 103
  XADgetx ................................................................. 104
  XADgety ................................................................. 105
  XADgetw ................................................................. 106
  XADgeth ................................................................. 107
  XADrefresh ............................................................... 108
  XADsetfocus ............................................................. 109
  XADsetpos ............................................................... 110
  XADsettext ............................................................. 111
  XADsetvisible ........................................................... 112
  XADgeteventtext ........................................................ 113

8 Generic events ................................................................. 114
  XAD_EVENT_KEYDOWN .................................................. 114
  XAD_EVENT_KEYUP ..................................................... 115
  XAD_EVENT_MOUSE_LEFTDOWN ......................................... 115
  XAD_EVENT_MOUSE_LEFTUP ............................................. 115
  XAD_EVENT_MOUSE_MOVED .............................................. 115
  XAD_EVENT_MOUSE_RIGHTDOWN ......................................... 115
  XAD_EVENT_MOUSE_RIGHTUP ............................................ 115

9 Utility routines ............................................................... 116
  XADid ................................................................. 117
  XADsavescreenshot ...................................................... 118
  XADseteventcallback .................................................... 119
  XADhandleevents ........................................................ 120
  XADchoosefile .......................................................... 121
  XADpopupmenu ........................................................... 122

10 Example ................................................................. 123

Index ................................................................. 126
Chapter 1
XAD Applications

High level applications of Mosel and XAD

• Build an interactive personnel assignment optimization application.
• Visualize the performance in time of an asset portfolio model.
• Prototype a bin packing model.
• Cutting stock problems too difficult to understand? Use XAD to clear up the picture.
• Deploy a facility location optimization application, with interactive GIS functionality.
• Prototype and deploy a vehicle routing application.
• Visualize strategic capacity planning with a 30-year horizon.
• Take advantage of parallel computing and visualize concurrent optimization runs.
• Write a simple text editor in 20 lines or less of Mosel code.
• Build breakthrough journaling/monitoring features which record how users interact with your application.
• Quickly build a visualization platform for your data, results, or both.
• Build data input forms for your users.

Development with Mosel and XAD is contiguous and seamless. Write Mosel code to build the user interface as well as the mathematical optimization model, data input/output and pre/postprocessing. Save an order of magnitude of development time by not having to switch editors, compilers, data paths, or even developers. The Mosel environment combined with XAD enables fast and intuitive development.
Xpress Application Developer (XAD) is an extension of the Xpress-Mosel modeling and programming language. Xpress Application Developer extends the functionality of Mosel with a set of functions and procedures for creating standard user interfaces. As a result, Mosel can be used as a modeling and programming language for complete optimization application development, from the mathematical representation of a problem to developing the user interface.

Xpress Application Developer can save significant amounts of time when experimenting with an optimization problem because the OR practitioner no longer needs to interface Mosel with VB, or C++, or Java in order to build a user application. Most features needed for GUI-based application development are now available through the powerful yet easy to use abstractions in the Mosel language.

For example, the following window, (which is a copy of the Optimize matrix dialog in IVE):

![Optimize matrix dialog](image)

was created exclusively in Mosel, using the following Mosel XAD code:
A few notes on using XAD with Mosel:

XAD code may be placed anywhere in a Mosel model, however, the user is encouraged to separate the code implementing the visual functionality from other modeling/programming statements, as needed.

In order to run a model which uses mmxad.dso, the library must be present in the dso folder of the Xpress-MP installation (or pointed to be the MOSEL_DOS environment variable) and properly licensed through the license file. Xpress-IVE is not needed—XAD is an independent library which only requires the Mosel runtime library.
Chapter 3

GUI creation with XAD

An XAD user interface is composed of windows that can be opened or closed. Each window may contain any number of XAD objects such as lists, buttons, checkboxes, etc. Every window as well as every XAD object has a unique integer identifier (referred to as id) associated with it. The identifier is associated with the graphical object at creation and is henceforth used to refer to the respective object later in the program/model.

When the user interacts with an XAD object, events are generated. For example pressing a button generates an XAD_EVENT_PRESSED event; selecting an item in a list generates an XAD_EVENT_SELECTION event. Events are processed in a callback routine written by the user. The behavior of the user interface is determined by how the application responds to events. Any Mosel code can be written for dealing with an event, including XAD statements for altering the state of the user interface. This allows great flexibility in dealing with Mosel data as well as interacting with the GUI.

A simple example will demonstrate the principles of creating and managing an XAD user interface. Suppose we want to write a Mosel program which displays a window asking for a number that is needed later in the model:

![Simple input window example](image)

Let's examine the code:

```mosel
model simple
uses "xmade":

declarations
  id_win; id_textnumber; id_1numbrer; id_buttonok & id; store ids for later
  N: real; // the number we are seeking
end-declarations

// user interface creation
XCreateWindow [id_win, 50, 50, 300, 200, "Need a number to continue!"]
XCreateText [id_win, id_textnumber, 20, 50, 80, 40, "Enter a number!"]
XCreateInput [id_win, id_1number, 100, 50, 100, 20, ""]
XCreateButton [id_win, id_buttonok, 110, 100, 100, 24, "OK"]
```

First we create a window and assign the id id_win to it. Then we create three XAD objects: a descriptive text, an input field and a button, each with a different id. The ids should be longer rather than shorter and they should provide type information to make the model more readable and maintainable.
The event handler is the core of an XAD program. All the interaction between the user and the GUI is reflected through the event handler. The event handler is a callback procedure that takes two arguments of type integer. XAD will call this procedure when an event occurs. The arguments are:

id: integer  The id of the XAD object that generated the event

event: integer  A number which denotes an event (e.g. XAD_EVENT_PRESSED)

Note that some events such as key presses and list selections carry textual information. If this information is needed, it can be retrieved in the event handler using the routine XADgeteventtext:

string

In our simple case, if the object id_buttonok is pressed then we close the window. When the window is closed, the text currently in the input object id_inputnumber is converted to a real number and assigned to N. Thirdly, for improved user interaction, we set the focus on the input object as soon as the window is opened.

Finally:

/event handler
proceudre guilevents(id:integer,event:integer)
    if id=id_buttonok and event=XAD_EVENT_PRESSED then
        XADwindowClose(id_win)
    elif id=id_win and event=XAD_EVENT_WINDOW_CLOSED then
        N=real(XADinputgettext(id_inputnumber))
    elsif id=id_win and event=XAD_EVENT_WINDOW_OPENED then
        XADsetfocus(id_inputnumber)
    end-if
end-procedure

The procedure which handles the events (the event handler) must be registered with XAD by calling XADseteventcallback with the event handler procedure name as the sole argument. Once the event handler is in place we can open (show) the window. When closing the window (by pressing the OK button or clicking the close button or pressing Esc), Mosel will resume its execution from the statement immediately following XADwindowopen.
Chapter 4

Lifetime of XAD objects

An XAD object will exist until the model ends or until it is destroyed using `XADdestroy`. The object-specific routines (listed in the reference part of this document) can only operate on an object when the window containing the object is active (or, if the object is a window, only while the window is active). For this reason, the event `XAD_EVENT_WINDOW_OPENED` is of particular interest: it is the only chance to operate on the objects in a window before the window begins interacting with the user.

There are two distinct ways in which windows may behave:

1. When a window is opened (using `XADwindowopen`), the window takes control of the Mosel program. Until the window is closed (using `XADwindowclose`) the only way to execute code is from the event handler (directly, or by calling a subroutine). The statement following `XADwindowopen` will only be executed after the user has closed the window. This behavior can be used when user input is necessary before the program can continue or when the user interface is meant to control the Mosel program. This type of window is referred to as a *modal dialog*.

2. When a window is shown (using `XADwindowshow`), the window is displayed and Mosel continues immediately. A shown window should be used as an auxiliary window for monitoring program state, to display progress, etc. When the monitoring is complete (e.g. at the end of the Mosel run), the window may be hidden, using `XADwindowhide`. If the window is not hidden before the Mosel run ends, it will persist until the model is unloaded from memory (in Xpress-IVE this only happens when compiling or running a model). This type of window is referred to as a *modeless dialog*.

It is important to make a correct choice between the two behaviors. Experiment until the distinction is well understood, and keep in mind that for most purposes a window should be opened in order to allow it to take control of the program execution by generating events.
Chapter 5
Events

When the user operates on a window or object in a window, the action is reported through an event. The event handler callback procedure can be used to respond to such events. For example, when the user presses a button, the event handler callback procedures is called with two arguments: an integer representing the id of the button that triggered the event and another integer representing the event code XAD_EVENT_PRESSED. Or, when the user resizes a window by dragging its margins, the event is reported as a call to the event handler callback with the window id and XAD_EVENT_WINDOW_RESIZED as arguments.
Chapter 6

XAD objects reference

XAD provides functionality to work with the following graphical objects:

- Browser (6.1)
- Button (6.2)
- Canvas (6.3)
- Check button (6.4)
- Drop list (6.5)
- Editor (6.6)
- Group (6.7)
- Input (6.8)
- List (6.9)
- Multilist (6.10)
- Progress bar (6.11)
- Radio button (6.12)
- Scroll bar (6.13)
- Tab (6.14)
- Text (6.15)
- Tree (6.16)
- Window (6.17)

6.1 Browser

An Internet Explorer-based web browser which can display any webpage.
Specific subroutines
   XADcreatebrowser, XADbrowsergoto

Specific events
   None

6.2 Button

A regular push button.

Specific subroutines
   XADcreatebutton

Specific events
   XAD_EVENT_PRESSED

6.3 Canvas

An intuitive surface for drawing anything using XAD.

Specific subroutines
   XADcreatecanvas, XADcanvasdrawarc, XADcanvasdrawbox, XADcanvasdrawchord, XADcanvasdrawellipse, XADcanvasdrawimage, XADcanvasdrawline, XADcanvasdrawpie, XADcanvasdrawpoint, XADcanvasdrawpolygon, XADcanvasdrawrectangle, XADcanvasdrawtext, XADcanvaserase, XADcanvasmap, XADcanvasrefresh, XADcanvassaveimage, XADcanvasunmap

Specific events
   None

Note  Working with colors. XAD recognizes the following color constants: XAD_BLACK, XAD_BLUE, XAD_CYAN, XAD_GREEN, XAD_MAGENTA, XAD_ORANGE, XAD_RED, XAD_WHITE, XAD_YELLOW Use XADcolor to create any other color based on its red, green and blue components.
6.4 Check button

A button with two states: checked and unchecked. It is independent of any other objects.

Specific subroutines

XADcreatecheck, XADcheckgetstate, XADchecksetstate

Specific events

XAD_EVENT_PRESSED

6.5 Drop list

A sortable list of strings, numbers, etc., which can be expanded or collapsed.

Specific subroutines

XADcreatedroplist, XADdroplistadd, XADdroplistgetsel, XADdroplistselect, XADdroplistshow

Specific events

XAD_EVENT_SELECTION

Note

A list object generates an XAD_EVENT_SELECTION event when the user changes the selection. In order to find out which item was selected, call the function XADgeteventtext: string.

6.6 Editor

A field for editing multi-line text.
Specific subroutines
XADcreateeditor, XADeditoraddtext, XADeditorgettext, XADeditorload, XADeditorsave, XADeditorsettext

Specific events
XAD_EVENT_CHANGED

6.7 Group
A thin frame surrounding a group of related objects.

Specific subroutines
XADcreategroup

Specific events
None

6.8 Input
A single line input field.
Specific subroutines
   XADcreateinput, XADinputgettext, XADinputsettext

Specific events
   XAD_EVENT_CHANGED

### 6.9 List

A sortable list of strings, numbers, etc.

![A simple window](image)

Specific subroutines
   XADcreatelistent, XADlistadd, XADlistgetsel, XADlistselect, XADlistshow

Specific events
   XAD_EVENT_SELECTION

**Note**  A list object generates an XAD_EVENT_SELECTION event when the user changes the selection. In order to find out which item was selected, call the function XADgeteventtext: string.

### 6.10 List with multiple columns

A sortable list of strings, numbers, etc. which can display multidimensional data.

![A simple window](image)

Specific subroutines
   XADcreatemultilist, XADmultilistrefresh, XADmultilistsetcolname,
   XADmultilistsetsize, XADmultilistsettext, XADmultilistshow

Specific events
   None
6.11 Progress bar

A visual progress indicator.

Specific subroutines
XADcreateprogress, XADprogressset

Specific events
None

6.12 Radio button

A button with two states: checked and unchecked. It can be used for mutually exclusive choices.

Specific subroutines
XADcreateradio, XADradiogetstate, XADradiosetstate

Specific events
XAD_EVENT_PRESSED

6.13 Scroll bars

A vertical or horizontal scrollbar for controlling position in a large document.
Specific subroutines
XADcreatescrollbar, XADscrollbargetpos, XADscrollbarset

Specific events
XAD_EVENT_CHANGED

Note A scrollbar object generates an XAD_EVENT_CHANGED event when the user interacts with it. Use XADscrollbargetpos to obtain the new position.

6.14 Tab selectors

A notebook-style object for choosing among general categories (of other objects, usually).

Specific subroutines
XADcreatetab, XADtabgettab, XADtabsettab

Specific events
XAD_EVENT_SELECTION

Note A tab object generates an XAD_EVENT_SELECTION event when the user changes the selection. In order to find out which tab was selected, call the function XADgeteventtext: string or use XADtabgettab.

6.15 Text

A multiline label for displaying text information.
Specific subroutines
XADcreatetext, XADtextaddtext, XADtextgettext, XADtextsettext

Specific events
None

6.16 Tree

A hierarchical tree display.

Specific subroutines
XADcreatetree, XADtreeadd, XADtreereset, XADtreeexpand

Specific events
XAD_EVENT_SELECTION

6.17 Window

The most important object; brings together all the other types of objects and allows interaction with the user.
Specific subroutines

- XADcreatewindow
- XADwindowaddmenu
- XADwindowclose
- XADwindowhide
- XADwindowkeep
- XADwindowopen
- XADwindowsettimer
- XADwindowshow

Specific events

- XAD_EVENT_MENU
- XAD_EVENT_TIMER
- XAD_EVENT_WINDOW_CLOSED
- XAD_EVENT_WINDOW_CLOSING
- XAD_EVENT_WINDOW_HIDDEN
- XAD_EVENT_WINDOW_MOVED
- XAD_EVENT_WINDOW_OPENED
- XAD_EVENT_WINDOW_RESIZED
- XAD_EVENT_WINDOW_SHOWN

### 6.18 Subroutines specific to objects

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XADbrowsergoto</td>
<td>Open the given URL in the browser.</td>
<td>88</td>
</tr>
<tr>
<td>XADcanvasdrawarc</td>
<td>Draw an elliptical arc on a canvas.</td>
<td>81</td>
</tr>
<tr>
<td>XADcanvasdrawbox</td>
<td>Draw a box on a canvas.</td>
<td>70</td>
</tr>
<tr>
<td>XADcanvasdrawchord</td>
<td>Draw an elliptical chord on a canvas.</td>
<td>82</td>
</tr>
<tr>
<td>XADcanvasdrawellipse</td>
<td>Draw an ellipse on a canvas.</td>
<td>71</td>
</tr>
<tr>
<td>XADcanvasdrawimage</td>
<td>Draw an image from file.</td>
<td>74</td>
</tr>
<tr>
<td>XADcanvasdrawline</td>
<td>Draw a line on a canvas.</td>
<td>76</td>
</tr>
<tr>
<td>XADcanvasdrawpie</td>
<td>Draw an elliptical pie slice on a canvas.</td>
<td>80</td>
</tr>
<tr>
<td>XADcanvasdrawpoint</td>
<td>Draw a point on a canvas.</td>
<td>77</td>
</tr>
<tr>
<td>XADcanvasdrawpolygon</td>
<td>Draw a polygon on a canvas.</td>
<td>79</td>
</tr>
<tr>
<td>XADcanvasdrawrectangle</td>
<td>Draw a rectangle on a canvas.</td>
<td>78</td>
</tr>
<tr>
<td>XADcanvasdrawtext</td>
<td>Draw text on a canvas.</td>
<td>83</td>
</tr>
<tr>
<td>XADcanvaserase</td>
<td>Erase a canvas.</td>
<td>72</td>
</tr>
<tr>
<td>XADcanvasmap</td>
<td>Map the coordinate space of a canvas.</td>
<td>84</td>
</tr>
<tr>
<td>XADcanvasrefresh</td>
<td>Redraw a canvas.</td>
<td>73</td>
</tr>
<tr>
<td>XADcanvassaveimage</td>
<td>Save an image to memory (to avoid loading a file repeatedly).</td>
<td>75</td>
</tr>
<tr>
<td>XADcanvasunmap</td>
<td>Revert to default mapping for a canvas.</td>
<td>85</td>
</tr>
<tr>
<td>XADcheckgetstate</td>
<td>Get the state of a check button.</td>
<td>43</td>
</tr>
<tr>
<td>XADchecksetstate</td>
<td>Set the state of a check button.</td>
<td>42</td>
</tr>
<tr>
<td>XADcolor</td>
<td>Create a color value.</td>
<td>86</td>
</tr>
<tr>
<td>XADcreatebrowser</td>
<td>Create a browser.</td>
<td>87</td>
</tr>
<tr>
<td>XADcreatebutton</td>
<td>Create a button object.</td>
<td>31</td>
</tr>
<tr>
<td>XADcreatecanvas</td>
<td>Create a canvas object.</td>
<td>69</td>
</tr>
<tr>
<td>XADcreatecheck</td>
<td>Create a check button.</td>
<td>41</td>
</tr>
<tr>
<td>XADcreatedroplist</td>
<td>Create a droplist object.</td>
<td>53</td>
</tr>
<tr>
<td>XADcreateeditor</td>
<td>Create an editor object.</td>
<td>35</td>
</tr>
<tr>
<td>XADcreategroup</td>
<td>Create a group object.</td>
<td>47</td>
</tr>
<tr>
<td>XADcreateinput</td>
<td>Create an input object.</td>
<td>32</td>
</tr>
<tr>
<td>XADcreatelist</td>
<td>Create a list object.</td>
<td>48</td>
</tr>
</tbody>
</table>
XADcreatemultilist Create a multilist object.  

XADcreateprogress Create a progress bar.  

XADcreateradio Create a radio button.  

XADcreatescrollbar Create a scrollbar object.  

XADcreatetab Create a tab selector object.  

XADcreatetext Create a text object.  

XADcreatetree Create a tree object.  

XADcreatetree Create a text object.  

XADdroplistadd Add an item to a droplist  

XADdroplistgetsel Get the selected item from a droplist.  

XADdroplistselect Select a droplist item  

XADdroplistshow Show a droplist.  

XADeditoraddtext Add text to an editor.  

XADeditorgettext Get the text from an editor.  

XADeditorload Load a file into an editor.  

XADeditorsave Save editor contents into a file.  

XADeditorsettext Set the text of an editor.  

XADinputgettext Get the text of an input object.  

XADinputsettext Set the text of an input object.  

XADlistadd Add an item to a list  

XADlistgetsel Get the selected item from a list.  

XADlistselect Select a list item  

XADlistshow Show a list.  

XADmultilistrefresh Update the visual display (usually after many XADmultilistsettext operations).  

XADmultilistsetcolname Set multilist column names.  

XADmultilistsetsize (Re)set the size of a multilist.  

XADmultilistsettext Set a multilist item.  

XADmultilistshow Load and display multi-dimensional arrays or a set in a multilist.  

XADprogressset Set the progress state.  

XADRadiogetstate Get the state of a radio button.  

XADRadiosetstate Get the state of a radio button.  

XADscrollbargetpos Obtain the current position of the scrollbar.  

XADscrollbarset Set scrollbar characteristics.  

XADtabgettab Get the current tab selection.  

XADtabsettab Select a given tab.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XADtextaddtext</td>
<td>Add text to a text object.</td>
<td>28</td>
</tr>
<tr>
<td>XADtextgettext</td>
<td>Get the text of a text object.</td>
<td>30</td>
</tr>
<tr>
<td>XADtextsettext</td>
<td>Set text of a text object.</td>
<td>29</td>
</tr>
<tr>
<td>XADtreeadd</td>
<td>Add a branch to a tree</td>
<td>93</td>
</tr>
<tr>
<td>XADtreeexpand</td>
<td>Expands a tree branch</td>
<td>95</td>
</tr>
<tr>
<td>XADtreereset</td>
<td>Clears the content of a tree</td>
<td>94</td>
</tr>
<tr>
<td>XADwindowaddmenu</td>
<td>Add a dropdown menu to the window.</td>
<td>26</td>
</tr>
<tr>
<td>XADwindowclose</td>
<td>Close a window.</td>
<td>21</td>
</tr>
<tr>
<td>XADwindowhide</td>
<td>Hide a window.</td>
<td>23</td>
</tr>
<tr>
<td>XADwindowkeep</td>
<td>Keep a window.</td>
<td>24</td>
</tr>
<tr>
<td>XADwindowopen</td>
<td>Open a window.</td>
<td>20</td>
</tr>
<tr>
<td>XADwindowsettimer</td>
<td>(Re)set a timer.</td>
<td>25</td>
</tr>
<tr>
<td>XADwindowshow</td>
<td>Display a window.</td>
<td>22</td>
</tr>
</tbody>
</table>
XADcreatewindow

**Purpose**
Create a window

**Synopsis**
procedure XADcreatewindow(id:integer, x:integer, y:integer, w:integer,
                           h:integer, name:string)

**Arguments**
- **id**: Arbitrary unique id of the window
- **x**: x coordinate (from left of screen) of the window in pixels
- **y**: y coordinate (from top of screen) of the window in pixels
- **w**: Width of the window in pixels
- **h**: Height of the window in pixels
- **name**: The title of the window

**Related topics**
XADwindowopen, XADwindowclose, XADwindowshow, XADwindowhide, XADwindowkeep, XADwindowsettimer
XADwindowopen

Purpose
Open a window.

Synopsis
procedure XADwindowopen(id:integer)

Argument
  id Window identifier

Further information
Displays the window and surrenders execution control to its event handler callback.

Related topics
XADwindowclose
**XADwindowclose**

**Purpose**
Close a window.

**Synopsis**

```pascal
procedure XADwindowclose(id:integer)
```

**Argument**

- `id` Window identifier

**Further information**

Destroys an *opened* window.

**Related topics**

- XADwindowopen
XADwindowshow

Purpose
Display a window.

Synopsis
procedure XADwindowshow(id:integer)

Argument
id       Window identifier

Further information
Displays the window and execution resumes immediately.

Related topics
XADwindowhide
XADwindowhide

Purpose
Hide a window.

Synopsis
procedure XADwindowhide(id:integer)

Argument
  id     Window identifier

Further information
Destroys a shown window.

Related topics
  XADwindowshow
XADwindowkeep

**Purpose**
Keep a window.

**Synopsis**
procedure XADwindowkeep

**Further information**
Should be used only when dealing with the `XAD_EVENT_WINDOW_CLOSING` event, if the intent is to cancel the closing of the window. For example, if an exit confirmation dialog is shown and the user changes his mind, call this procedure to prevent the window from closing.

**Related topics**
XADwindowhide
XADwindowsettimer

Purpose
(Re)set a timer.

Synopsis
procedure XADwindowsettimer(id:integer, msec:integer)

Arguments
id Window identifier
msec Timer interval in milliseconds

Further information
Sets or resets a timer that generates XAD_EVENT_TIMER events. If msec is a positive integer, the event will be generated every msec milliseconds. If msec is 0 or negative, the timer is reset (timer events are no longer generated)
XADwindowaddmenu

Purpose
Add a dropdown menu to the window.

Synopsis
procedure XADwindowaddmenu(id:integer, title:string, items:string)
procedure XADwindowaddmenu(id:integer, title:string, itemset: set of string)

Arguments
id   Window identifier
title Menu title (e.g. File, Edit)
items Comma-separated list of strings representing the menu items
itemset Set of strings representing the menu items

Example
XADwindowaddmenu(id_win,"","") !pass empty string to clear existing menu, if any
XADwindowaddmenu(id_win,"&File","&New,&Open,&Save,XADseparator,&Exit")
XADwindowaddmenu(id_win,"&Help","&Help on MyApp,&About...")

Further information

Respond to menu selections by checking the XAD_EVENT_MENU event.
Use the ampersand character & to create a Alt+letter hotkey for each menu item.
Use the string "XADseparator" to insert a line separator in the menu.
Pass an empty title to reset and destroy the menu. Then rebuild the entire menu.
XADcreatetext

**Purpose**
Create a text object.

**Synopsis**
```pascal
procedure XADcreatetext(wid:integer, id:integer, x:integer, y:integer,
                        w:integer, h:integer, name:string)
```

**Arguments**
- `wid` id of the window containing the text
- `id` Text identifier
- `x` x coordinate (from left of window) of the text in pixels
- `y` y coordinate (from top of window) of the text in pixels
- `w` Width of the text in pixels
- `h` Height of the text in pixels
- `text` Contents of the text object (use \r\n to break lines)

**Related topics**
- XADtextaddtext, XADtextgettext, XADtextsettext
XADtextaddtext

Purpose
Add text to a text object.

Synopsis
procedure XADtextaddtext(id:integer, text:string)

Arguments
id Text identifier
text Text to be added

Further information
Appends the given text to the object.

Related topics
XADtextgettext, XADtextsettext
XADtextsettext

Purpose
Set text of a text object.

Synopsis
procedure XADtextsettext(id:integer, text:string)

Arguments
id Text identifier
text New text

Further information
Replaces the text of the object with the new text.

Related topics
XADtextaddtext, XADtextgettext
**XADtextgettext**

**Purpose**
Get the text of a text object.

**Synopsis**

```haskell
function XADtextgettext(id:integer):string
```

**Argument**
- id: Text identifier

**Return value**
- Text of the object.

**Further information**
Obtains the text currently shown by the object.

**Related topics**
- XADtextaddtext, XADtextsettext
XADcreatebutton

Purpose
Create a button object.

Synopsis
procedure XADcreatebutton(wid:integer, id:integer, x:integer, y:integer, w:integer, h:integer, name:string)

Arguments
   wid  id of the window containing the button
   id   Button identifier
   x    x coordinate (from left of window) of the button in pixels
   y    y coordinate (from top of window) of the button in pixels
   w    Width of the button in pixels
   h    Height of the button in pixels
   name The name of the button
XADcreateinput

Purpose
Create an input object.

Synopsis
procedure XADcreateinput(wid:integer, id:integer, x:integer, y:integer,
w:integer, h:integer, text:string)

Arguments
wid id of the window containing the input
id Input identifier
x x coordinate (from left of window) of the input in pixels
y y coordinate (from top of window) of the input in pixels
w Width of the input in pixels
h Height of the input in pixels
text Initial content of the input object

Related topics
XADinputgettext, XADinputsettext
XADinputsettext

Purpose
Set the text of an input object.

Synopsis
procedure XADinputsettext(id:integer, text:string)

Arguments
id Input identifier
text New text

Further information
Replaces the text of the object with the new text.

Related topics
XADinputgettext
XADinputgettext

Purpose
Get the text of an input object.

Synopsis
function XADinputgettext(id:integer):string

Argument
id       Input identifier

Return value
Text of the object.

Further information
Obtains the text currently shown by the object.

Related topics
XADinputsettext
XADcreateeditor

**Purpose**
Create an editor object.

**Synopsis**

```pascal
procedure XADcreateeditor(wid:integer, id:integer, x:integer, y:integer,
                           w:integer, h:integer, text:string)
```

**Arguments**

- **wid**  
id of the window containing the editor
- **id**  
Editor identifier
- **x**  
x coordinate (from left of window) of the editor in pixels
- **y**  
y coordinate (from top of window) of the editor in pixels
- **w**  
Width of the editor in pixels
- **h**  
Height of the editor in pixels
- **text**  
Contents of the editor object (use `\r\n` to break lines)

**Related topics**

- XADeditoraddtext
- XADeditorload
- XADeditorsave
- XADeditorsettext
- XADeditorgettext
XADeditoraddtext

Purpose
Add text to an editor.

Synopsis
procedure XADeditoraddtext(id:integer, text:string)

Arguments
id Editor identifier
text Text to be added

Further information
Appends the given text to the object.

Related topics
XADeditorsettext, XADeditorgettext
XADeditorload

**Purpose**
Load a file into an editor.

**Synopsis**
procedure XADeditorload(id:integer, file:string)

**Arguments**
- *id*    Editor identifier
- *file*  File name

**Further information**
Load the specified *file* into the editor, replacing its previous content.

**Related topics**
XADeditorsave
**XADEditorsave**

**Purpose**
Save editor contents into a file.

**Synopsis**
```
procedure XADEditorsave(id:integer, file:string)
```

**Arguments**
- `id` Editor identifier
- `file` File name

**Further information**
Saves the contents of the editor to the specified `file`.

**Related topics**
- XADEditorload
XADeditorsettext

Purpose
Set the text of an editor.

Synopsis
procedure XADeditorsettext(id:integer, text:string)

Arguments
id Editor identifier
text New text

Further information
Replaces the text of the object with the new text.

Related topics
XADeditoraddtext, XADeditorgettext
**XADeditorgettext**

**Purpose**
Get the text from an editor.

**Synopsis**
function XADeditorgettext(id:integer):string

**Argument**
- id Editor identifier

**Return value**
Text of the object.

**Further information**
Obtains the text currently shown by the object.

**Related topics**
XADeditoraddtext, XADeditorsettext,
XADcreatecheck

Purpose
Create a check button.

Synopsis
procedure XADcreatecheck(wid:integer, id:integer, x:integer, y:integer,
w:integer, h:integer, name:string)

Arguments
wid id of the window containing the check
id Check identifier
x x coordinate (from left of window) of the check in pixels
y y coordinate (from top of window) of the check in pixels
w Width of the check in pixels
h Height of the check in pixels
name Description of the check object

Related topics
XADchecksetstate, XADcheckgetstate
XADchecksetstate

Purpose
Set the state of a check button.

Synopsis
procedure XADchecksetstate(id:integer, state:boolean)

Arguments

<table>
<thead>
<tr>
<th>id</th>
<th>Check identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Button state.</td>
</tr>
<tr>
<td></td>
<td>true  checked</td>
</tr>
<tr>
<td></td>
<td>false unchecked</td>
</tr>
</tbody>
</table>

Further information
Sets the state of the check to checked or unchecked.

Related topics
XADcheckgetstate
XADcheckgetstate

**Purpose**
Get the state of a check button.

**Synopsis**

```plaintext
function XADcheckgetstate(id:integer):boolean
```

**Argument**

- `id` Check identifier

**Return value**

- `true` Button is checked
- `false` Button is unchecked

**Further information**
Obtains the state of the check (`true` for checked, `false` for unchecked).

**Related topics**

- `XADchecksetstate`
**XADcreateradio**

**Purpose**
Create a radio button.

**Synopsis**

```
procedure XADcreateradio(wid:integer, id:integer, x:integer, y:integer,
w:integer, h:integer, name:string)
```

**Arguments**

- `wid`: id of the window containing the radio
- `id`: Radio identifier
- `x`: x coordinate (from left of window) of the radio in pixels
- `y`: y coordinate (from top of window) of the radio in pixels
- `w`: Width of the radio in pixels
- `h`: Height of the radio in pixels
- `name`: Name of the radio button object

**Related topics**

`XADradiosetstate`, `XADradiogetstate`
XADradiosetstate

Purpose
Get the state of a radio button.

Synopsis
procedure XADradiosetstate(id:integer, state:boolean)

Arguments
id Radio identifier
state Button state.
  true selected
  false unselected

Further information
Sets the state of the radio to selected (true) or unselected (false).

Related topics
XADradiogetstate
XADradiogetstate

Purpose

Get the state of a radio button.

Synopsis

function XADradiogetstate(id:integer):boolean

Arguments

id Radio identifier

Return value

true Radio button is selected
false Radio button is unselected

Further information

Obtains the state of the radio (true for selected, false for unselected).

Related topics

XADradiosetstate
XADcreategroup

Purpose
Create a group object.

Synopsis
procedure XADcreategroup(wid:integer, id:integer, x:integer, y:integer,
                         w:integer, h:integer, name:string)

Arguments
wid   id of the window containing the group
id    Group identifier
x     x coordinate (from left of window) of the group in pixels
y     y coordinate (from top of window) of the group in pixels
w     Width of the group in pixels
h     Height of the group in pixels
name  Description of the group
XADcreatelis

Purpose
Create a list object.

Synopsis
procedure XADcreatelis(wid:integer, id:integer, x:integer, y:integer,
 w:integer, h:integer, name:string)
procedure XADcreatelis(wid:integer, id:integer, x:integer, y:integer,
 w:integer, h:integer, name:string, sorted:boolean)

Arguments
wid id of the window containing the list
id List identifier
x x coordinate (from left of window) of the list in pixels
y y coordinate (from top of window) of the list in pixels
w Width of the list in pixels
h Height of the list in pixels
name String containing comma-separated list items
sorted If true, the list will be sorted and remain sorted after adding elements

Related topics
XADlistadd, XADlistgetsel, XADlistshow
**XADlistadd**

**Purpose**
Add an item to a list

**Synopsis**

```plaintext
procedure XADlistadd(id:integer, item:string)
```

**Arguments**

- **id** List identifier
- **item** Item to add

**Further information**

Appends the *item* to the list.

**Related topics**

- XADlistgetsel
- XADlistshow
XADlistgetsel

Purpose
Get the selected item from a list.

Synopsis
function XADlistgetsel(id:integer):string

Arguments
id List identifier

Return value
Selected item.

Further information
Obtains the item currently selected.

Related topics
XADlistadd, XADlistshow, XADlistselect
XADlistselect

Purpose
Select a list item

Synopsis
procedure XADlistselect(id:integer, item:string)

Arguments
  id    List identifier
  item  Item to select

Further information
If found, selects the item.

Related topics
XADlistgetsel, XADlistshow
**XADlistshow**

**Purpose**
Show a list.

**Synopsis**
procedure XADlistshow(id:integer, items:set)

**Arguments**
id List identifier
items Items to display

**Further information**
Fills the list with the contents of the given set of items. Mosel will try to infer the type of the set and display accordingly. For example showing a set of mpvar will display the solution values and reduced costs for each item.

**Related topics**
XADlistadd, XADlistget.sel
**XADcreatedroplist**

**Purpose**
Create a droplist object.

**Synopsis**

```plaintext
procedure XADcreatedroplist(wid:integer, id:integer, x:integer, y:integer, w:integer, h:integer, name:string)
procedure XADcreatedroplist(wid:integer, id:integer, x:integer, y:integer, w:integer, h:integer, name:string, sorted:boolean)
```

**Arguments**

- `wid`: id of the window containing the droplist
- `id`: Droplist identifier
- `x`: x coordinate (from left of window) of the droplist in pixels
- `y`: y coordinate (from top of window) of the droplist in pixels
- `w`: Width of the droplist in pixels
- `h`: Height of the droplist in pixels
- `name`: String containing comma-separated droplist items
- `sorted`: If true, the droplist will be sorted and remain sorted after adding elements

**Related topics**

- `XADdroplistadd`, `XADdroplistgetsel`, `XADdroplistshow`
XADdroplistadd

Purpose
Add an item to a droplist

Synopsis
procedure XADdroplistadd(id:integer, item:string)

Arguments
id   Droplist identifier
item Item to add

Further information
Appends the item to the droplist.

Related topics
XADdroplistgetsel, XADdroplistshow
XADdroplistgetsel

Purpose
Get the selected item from a droplist.

Synopsis
function XADdroplistgetsel(id:integer):string

Arguments
id  Droplist identifier

Return value
Selected item.

Further information
Obtains the item currently selected.

Related topics
XADdroplistadd, XADdroplistshow, XADdroplistselect
XADdroplistselect

Purpose
Select a droplist item

Synopsis
procedure XADdroplistselect(id:integer, item:string)

Arguments
id        Droplist identifier
item      Item to select

Further information
If found, selects the item.

Related topics
XADdroplistgetsel, XADdroplistshow
XADdroplistshow

Purpose
Show a droplist.

Synopsis
procedure XADdroplistshow(id:integer, items:set)

Arguments
id Droplist identifier
items Items to display

Further information
Fills the droplist with the contents of the given set of items. Mosel will try to infer the type of the set and display accordingly. For example showing a set of mpvar will display the solution values and reduced costs for each item.

Related topics
XADdroplistadd, XADdroplistgetsel
XADcreateprogress

**Purpose**
Create a progress bar.

**Synopsis**
```pascal
procedure XADcreateprogress(wid:integer, id:integer, x:integer,
                           y:integer, w:integer, h:integer)
```

**Arguments**
- `wid` id of the window containing the progress
- `id` Progress identifier
- `x` x coordinate (from left of window) of the progress in pixels
- `y` y coordinate (from top of window) of the progress in pixels
- `w` Width of the progress in pixels
- `h` Height of the progress in pixels

**Related topics**
- XADprogressset
XADprogressset

Purpose
Set the progress state.

Synopsis
procedure XADprogressset(id:integer, minval:real, maxval:real, current:real)

Arguments
id      Progress identifier
minval  Lower bound
maxval  Upper bound
current  Current value

Further information
The progress will show how far current is between minval and maxval.
XADcreatetab

Purpose
Create a tab selector object.

Synopsis
procedure XADcreatetab(wid:integer, id:integer, x:integer, y:integer,
w:integer, h:integer, tabs:set of string)

Arguments
wid   id of the window containing the tab
id    Tab identifier
x     x coordinate (from left of window) of the tab in pixels
y     y coordinate (from top of window) of the tab in pixels
w     Width of the tab in pixels
h     Height of the tab in pixels
tabs  Set of strings containing the possible choices
**XADtabgettab**

**Purpose**
Get the current tab selection.

**Synopsis**
```
function XADtabgettab(id:integer):string
```

**Argument**
- id : Tab identifier

**Return value**
- Selected item.
XADtabsettab

Purpose
Select a given tab.

Synopsis
procedure XADtabsettab(id:integer, tab:string)

Arguments
  id  Tab identifier
  tab tab to be selected
XADcreatemultilist

**Purpose**
Create a multilist object.

**Synopsis**

```pascal
procedure XADcreatemultilist(wid:integer, id:integer, name:string,
                              x:integer, y:integer, w:integer, h:integer)
```

**Arguments**

- **wid**: id of the window containing the multilist
- **id**: Multilist identifier
- **x**: x coordinate (from left of window) of the multilist in pixels
- **y**: y coordinate (from top of window) of the multilist in pixels
- **w**: Width of the multilist in pixels
- **h**: Height of the multilist in pixels

**Related topics**

- XADmultilistshow
- XADmultilistsetsize
- XADmultilistsetcolname
- XADmultilistsettext
**XADmultilistshow**

**Purpose**
Load and display multi-dimensional arrays or a set in a multilist.

**Synopsis**
procedure XADmultilistshow(id:integer, items: array)

**Arguments**
- **id**  Multilist identifier
- **items**  Comma-separated list of array names to display in the multilist, or just one array name.

**Example**

```
XADmultilistshow(id_multi, "Units, Salesforce, NetProfit")
```

**Further information**
Clears the content of the multilist and fills it with the given array of *items* (integers, reals, strings, booleans, mpvars or linets). If more than one array is given, they must have the exact same index sets and shape.

**Related topics**
XADmultilistsetsize, XADmultilistsetcolname, XADmultilistsettext
XADmultilistsetsize

**Purpose**
(Re)set the size of a multilist.

**Synopsis**

```plaintext
procedure XADmultilistsetsize(id:integer, rows:integer, columns:integer)
procedure XADmultilistsetsize(id:integer, rows:integer, columns:integer, 
callback:string)
```

**Arguments**

- `id`: Multilist identifier
- `rows`: Number of rows
- `columns`: Number of columns
- `callback`: Name of callback function for requesting element values

**Example 1**
A static multilist

```plaintext
procedure SetUpStaticList
    XADmultilistsetsize(id_multidynamic,10000,8)
    forall(i in 1..8) XADmultilistsetcolname(id_multidynamic,i,"Col "+i)
    forall(i in 1..10000,j in 1..8) XADmultilistsettext(id_multidynamic,i,j,""+i+","+j"
    XADmultilistrefresh(id_multidynamic)
end-procedure
```

**Example 2**
A dynamic multilist

```plaintext
function ElementGenerator(id:integer,row:integer,col:integer):string
    returned:=""+row+","+col !generate element based on row, col
end-function

procedure SetUpDynamicList
    ! set up the multilist to ask for its items only when they are needed
    ! a multilist with 10,000 rows and 8 columns which uses a callback
    XADmultilistsetsize(id_multidynamic,10000,8,"ElementGenerator")
    ! column names
    forall(i in 1..8) XADmultilistsetcolname(id_multidynamic,i,"Col "+i
end-procedure
```

**Further information**
Clears the content of the multilist and prepares to hold the given number of rows and columns.
Note that row and column indices start with 1.

**Related topics**

- XADmultilistsetcolname
- XADmultilistsettext
**XADmultilistsetcolname**

**Purpose**
Set multilist column names.

**Synopsis**
procedure XADmultilistsetcolname(id:integer, column:integer, name:string)

**Arguments**
- **id**
  Multilist identifier
- **column**
  Column index
- **name**
  Column name

**Further information**
The header of the *column* will display the new *name*.

**Related topics**
- XADmultilistsetsize
- XADmultilistsetcolname
- XADmultilistrefresh
XADmultilistsettext

Purpose
Set a multilist item.

Synopsis
procedure XADmultilistsettext(id:integer, row:integer, column:integer, text:string)

Arguments
id   Multilist identifier
column  Column index
name  Column name
text  New text of the item

Further information
The item at the given row and column will hold the new text.

Related topics
XADmultilistsetsize, XADmultilistsetcolname, XADmultilistrefresh
**XADmultilistrefresh**

**Purpose**
Update the visual display (usually after many XADmultilistsettext operations).

**Synopsis**
procedure XADmultilistrefresh(id:integer)

**Argument**
id Multilist identifier

**Related topics**
XADmultilistsetsize, XADmultilistsetcolname, XADmultilistsettext
XADcreatecanvas

Purpose
Create a canvas object.

Synopsis
procedure XADcreatecanvas(wid:integer, id:integer, x:integer, y:integer,
                          w:integer, h:integer)

Arguments
wid  id of the window containing the canvas
id   Canvas identifier
x    x coordinate (from left of window) of the object in pixels
y    y coordinate (from top of window) of the object in pixels
w    Width of the object in pixels
h    Height of the object in pixels

Related topics
XADcanvasdrawbox, XADcanvasdrawellipse, XADcanvaserase, XADcanvasrefresh,
XADcanvasdrawimage, XADcanvassaveimage, XADcanvasdrawline,
XADcanvasdrawpoint, XADcanvasdrawrectangle, XADcanvasdrawtext, XADcanvasmap,
XADcanvasunmap, XADcolor
**XADcanvasdrawbox**

**Purpose**
Draw a box on a canvas.

**Synopsis**
procedure XADcanvasdrawbox(id:integer, x:real, y:real, w:real, h:real, color1:integer, color2:integer)

**Arguments**
id   Canvas identifier
x   x coordinate (from left of canvas)
y   y coordinate (from top of canvas)
w   width of box
h   height of box
color1   Border color
color2   Fill color

**Further information**
Draws a solid rectangle of color color2 with a one-pixel margin of color color1 of width w and height h at coordinates (x,y).

**Related topics**
XADcanvasdrawline, XADcanvasdrawpoint, XADcanvasdrawrectangle, XADcanvasdrawtext, XADcolor
XADcanvasdrawellipse

Purpose
Draw an ellipse on a canvas.

Synopsis
procedure XADcanvasdrawellipse(id:integer, x:real, y:real, w:real,
h:real, color1:integer, color2:integer)

Arguments
id  Canvas identifier  
x  x coordinate (from left of canvas)  
y  y coordinate (from top of canvas)  
w  width of ellipse  
h  height of ellipse  
color1  Border color  
color2  Fill color

Further information
Draws a solid ellipse of color color2 with a one-pixel margin of color color1 of width w and height h at coordinates (x,y).

Related topics
XADcanvasdrawline, XADcanvasdrawpoint, XADcanvasdrawrectangle, XADcanvasdrawtext, XADcolor
XADcanvaserase

Purpose
Erase a canvas.

Synopsis
procedure XADcanvaserase(id:integer, color:integer)

Arguments
id Canvas identifier
color Color selection

Further information
Clears the contents of the canvas using the specified color

Related topics
XADcanvasrefresh
XADcanvasrefresh

Purpose
Redraw a canvas.

Synopsis
procedure XADcanvasrefresh(id: integer)

Argument
id Canvas identifier

Further information
Updates the visual content of the canvas. Simply drawing on a canvas does not update its appearance (to save time). Only call XADcanvasrefresh when necessary.

Related topics
XADcanvaserase
XADcanvasdrawimage

Purpose
Draw an image from file.

Synopsis
procedure XADcanvasdrawimage(id:integer, x:real, y:real, file: string)
procedure XADcanvasdrawimage(id:integer, x:real, y:real, w:integer,
                              h:integer, file: string)

Arguments
id   Canvas identifier
x    x coordinate
y    y coordinate
w    Width
h    Height
file  File name

Further information
Draws the image from the given file at coordinates (x, y). The file can be of type .bmp, .jpg, .gif, or .png. If height and width are given, then the image is rescaled to fit in a rectangle of width w and height h. If the string is "xadimg:imgname", a previously saved imgname (see above) will be drawn.

Related topics
XADcanvassaveimage
XADcanvassaveimage

Purpose
Save an image to memory (to avoid loading a file repeatedly).

Synopsis
procedure XADcanvassaveimage(id:integer, x:real, y:real, w:integer, h:integer, imgname: string)
procedure XADcanvassaveimage(id:integer, filename:string, imgname: string)

Arguments
id | Canvas identifier
x | x coordinate
y | y coordinate
w | Width
h | Height
filename | File containing an image (.bmp, .jpg, .gif, or .png)
imgname | Image name

Further information
Saves what is currently drawn at $x, y, w, h$ or the image in $filename$ under the identifier $imgname$. $imgname$ can then be drawn anywhere else using XADcanvasdrawimage.

Related topics
XADcanvasdrawimage
**XADcanvasdrawline**

**Purpose**
Draw a line on a canvas.

**Synopsis**

```plaintext
procedure XADcanvasdrawline(id:integer, x1:real, y1:real, x2:real, y2:real, color:integer)
procedure XADcanvasdrawline(id:integer, x1:real, y1:real, x2:real, y2:real, color:integer, width: integer)
```

**Arguments**

- **id**  
  Canvas identifier
- **x1**  
  x start coordinate
- **y1**  
  y start coordinate
- **x2**  
  x end coordinate
- **y2**  
  y end coordinate
- **color**  
  Color
- **width**  
  Width

**Further information**

Draws a line `width`-pixels wide from \((x1, y1)\) to \((x2, y2)\) with the given `color`. The default width is one pixel.

**Related topics**

- XADcanvasdrawbox
- XADcanvasdrawpoint
- XADcanvasdrawrectangle
- XADcanvasdrawtext
- XADcolor
XADcanvasdrawpoint

Purpose
Draw a point on a canvas.

Synopsis
procedure XADcanvasdrawpoint>(id:integer, x:real, y:real, color:integer)

Arguments
id  Canvas identifier
x   x coordinate
y   y coordinate
color  Color

Further information
Draws a pixel at (x,y) with the given color.

Related topics
XADcanvasdrawbox, XADcanvasdrawline, XADcanvasdrawrectangle,
XADcanvasdrawtext, XADcolor
XADcanvasdrawrectangle

Purpose
Draw a rectangle on a canvas.

Synopsis
procedure XADcanvasdrawrectangle>(id:integer, x:real, y:real, w:real, h:real, color:integer)

Arguments
id Canvas identifier
x x coordinate
y y coordinate
w Width
h Height
color Color

Further information
Draws a solid rectangle of width \(w\) and height \(h\) at coordinates \((x, y)\) with the given color.

Related topics
XADcanvasdrawbox, XADcanvasdrawline, XADcanvasdrawpoint, XADcanvasdrawtext, XADcolor
XADcanvasdrawpolygon

Purpose
Draw a polygon on a canvas.

Synopsis
procedure XADcanvasdrawpolygon(id:integer, xs:array of real, ys:array of real, color1:integer, color2:integer)

Arguments
id Canvas identifier
xs Array of x coordinates
ys Array of y coordinates
color1 Border color
color2 Fill color

Example
declarations
   POINTS=1..3
   xs,ys:array(POINTS) of real
end-declarations
xs:=[50, 100, 30]
ys:=[50, 200, 80]
...
XADcanvasdrawpolygon(id_canvas,xs,ys,XAD_RED,XAD_BLACK)

Further information
Draws a using x and y coordinates taken from the xs and ys arrays. xs and ys must be indexed by the same index set. See example.

Related topics
XADcanvasdrawbox, XADcanvasdrawline, XADcanvasdrawpoint, XADcanvasdrawtext, XADcolor
XADcanvasdrawpie

Purpose

Draw an elliptical pie slice on a canvas.

Synopsis

procedure XADcanvasdrawpie(id:integer, x:real, y:real, w:real, h:real,
start:real, end:real, color1:integer, color2:integer)

Arguments

id        Canvas identifier
x         x coordinate
y         y coordinate
w         Width of bounding rectangle
h         Height of bounding rectangle
start     Clockwise start percentage [0-100]
end       Clockwise end percentage [0-100]
color1    Border color
color2    Fill color

Example

declarations
  sizes:array(1..4) of real
  colors:array(1..4) of integer
  base:real
end-declarations

colors:=[XAD_RED,XAD_GREEN,XAD_MAGENTA,XAD_BLACK]
forall(o in 1..4) sizes(o):=random*35; ! between 0-35%
...

base:=0
forall(o in 1..4) do
  !circular pie
  XADcanvasdrawpie(id_canvas,100,150,100,100,base,sizes(o),colors(o),colors(o))
  base+=sizes(o)
end-do

Further information

Draws a pie slice bounded by the given rectangle. The "angle" of the slice must be between [0-100] (percentage) and is measured clockwise beginning at 12 o'clock. See example.

Related topics

XADcanvasdrawbox, XADcanvasdrawarc, XADcanvasdrawchord
XADcanvasdrawarc

Purpose
Draw an elliptical arc on a canvas.

Synopsis
procedure XADcanvasdrawarc(id:integer, x:real, y:real, w:real, h:real,
start:real, end:real, color:integer)

Arguments
id   Canvas identifier
x    x coordinate
y    y coordinate
w    Width of bounding rectangle
h    Height of bounding rectangle
start Clockwise start percentage [0-100]
end  Clockwise end percentage [0-100]
color Color

Further information
Draws an elliptical arc bounded by the given rectangle. The "angle" of the arc must be
between [0-100] (percentage) and is measured clockwise beginning at 12 o'clock.

Related topics
XADcanvasdrawbox, XADcanvasdrawpie, XADcanvasdrawchord
**XADcanvasdrawchord**

**Purpose**
Draw an elliptical chord on a canvas.

**Synopsis**
procedure XADcanvasdrawchord(id:integer, x:real, y:real, w:real, h:real,
start:real, end:real, color1:integer, color2:integer)

**Arguments**
- **id**: Canvas identifier
- **x**: x coordinate
- **y**: y coordinate
- **w**: Width of bounding rectangle
- **h**: Height of bounding rectangle
- **start**: Clockwise start percentage [0-100]
- **end**: Clockwise end percentage [0-100]
- **color1**: Border color
- **color2**: Fill color

**Example**

```plaintext
declarations
  sizes:array(1..4) of real
  colors:array(1..4) of integer
  base:real
end-declarations

colors:=[XAD_RED,XAD_GREEN,XAD_MAGENTA,XAD_BLACK]
forall(o in 1..4) sizes(o):=random*35; ! between 0-35%
...

base:=0
forall(o in 1..4) do
  !circular pie
  XADcanvasdrawchord(id_canvas,100,150,100,100,base,sizes(o),colors(o),colors(o))
  base+=sizes(o)
end-do
```

**Further information**
Draws a elliptical chord bounded by the given rectangle. The "angle" of the chord must be between [0-100] (percentage) and is measured clockwise beginning at 12 o'clock. See example.

**Related topics**
- XADcanvasdrawbox
- XADcanvasdrawarc
- XADcanvasdrawpie
XADcanvasdrawtext

Purpose
Draw text on a canvas.

Synopsis
procedure XADcanvasdrawtext(id:integer, x:real, y:real, text:string, color:integer)
procedure XADcanvasdrawtext(id:integer, x:real, y:real, text:string, color:integer, fontsize:integer, alignment:integer, fontname:string)

Arguments
id        Canvas identifier
x         x coordinate
y         y coordinate
text      Text
color     Color
fontsize   Font size
alignment  Vertical and horizontal alignment
fontname  Font name

Further information
Draws text with font fontname of size fontsize at coordinates (x,y) with the given color and alignment. If font name and size are not given then the output uses Arial 10pt. The alignment is the sum of one of XAD_CENTERH, XAD_LEFT, and XAD_RIGHT (horizontal alignment) plus one of XAD_CENTERV, XAD_TOP, and XAD_BOTTOM (vertical alignment). XAD_DEFAULT can be used instead to specify upper left alignment.

Related topics
XADcanvasdrawbox, XADcanvasdrawline, XADcanvasdrawpoint, XADcanvasdrawrectangle, XADcolor
XADcanvasmap

Purpose
Map the coordinate space of a canvas.

Synopsis
procedure XADcanvasmap(id:integer, x1:real, y1:real, x2:real, y2:real, x1new:real, y1new:real, x2new:real, y2new:real)

Arguments
id  Canvas identifier
x1  x start coordinate
y1  y start coordinate
x2  x end coordinate
y2  y end coordinate
x1new  New x start coordinate
y1new  New y start coordinate
x2new  New x end coordinate
y2new  New y end coordinate

Further information
Transforms the coordinate space of the canvas. Normally, (0,0) represents the upper left corner of the canvas and widths and heights are measured in pixels. By remapping the coordinate space, transformations no longer need to be applied to coordinates when drawing. For example, if the canvas has a width of 300 and a height of 200 and if we want to draw a graph from -1 to +1 on the x axis and -2 to +2 on the y axis in a region of the original canvas, we could write, say:

XADcanvasmap(id_canvas, 50, 150, 250, 50, -1, -2, 1, 2)

before plotting the points on the graph. Note that this transformation reverts the direction of the y axis and plots everything between (-1,1) and (-2,2) in the portion of the canvas between pixel coordinates (50,50) and (250,150). The following figure clarifies the effect of the XADcanvasmap call:

After calling this procedure, coordinates should be given in the red intervals. XAD will map them correctly onto the canvas, based on the black intervals.

Related topics
XADcanvasunmap
XADcanvasunmap

Purpose
Revert to default mapping for a canvas.

Synopsis
procedure XADcanvasunmap(id:integer)

Argument
   id     Canvas identifier

Further information
Reverts to default mapping of coordinates

Related topics
   XADcanvasmap
XADcolor

Purpose
Create a color value.

Synopsis
function XADcolor(red:real, green:real, blue:real):integer

Arguments
red      Intensity of red (between 0 and 255)
green    Intensity of green (between 0 and 255)
blue     Intensity of blue (between 0 and 255)

Return value
Color value.

Further information
Creates a color value based on intensities of red, green and blue.

Related topics
See Section 6.3 for a list of predefined color constants.
**XADcreatebrowser**

**Purpose**
Create a browser.

**Synopsis**

```pascal
procedure XADcreatebrowser(wid:integer, id:integer, x:integer, y:integer, w:integer, h:integer, url:string)
```

**Arguments**

- **wid** id of the window containing the browser
- **id** Browser identifier
- **x** x coordinate (from left of window) of the browser in pixels
- **y** y coordinate (from top of window) of the browser in pixels
- **w** Width of the browser in pixels
- **h** Height of the browser in pixels
- **url** URL to open when the object is created

**Related topics**

- XADbrowsergoto
XADbrowsergoto

Purpose
Open the given URL in the browser.

Synopsis
procedure XADbrowsergoto(id:integer, url:string)

Arguments
id Browser identifier
text The URL to visit.

Further information
Visits the given url.

Related topics
XADcreatebrowser
XADcreatescrollbar

Purpose
Create a scrollbar object.

Synopsis
procedure XADcreatescrollbar(wid:integer, id:integer, x:integer, y:integer, w:integer, h:integer, vertical:boolean)

Arguments
wid    id of the window containing the scrollbar
id     Scrollbar identifier
x      x coordinate (from left of window) of the scrollbar in pixels
y      y coordinate (from top of window) of the scrollbar in pixels
w      Width of the scrollbar in pixels
h      Height of the scrollbar in pixels
vertical  true=vertical; false=horizontal

Related topics
XADscrollbarset, XADscrollbargetpos
**XADscrollbargetpos**

**Purpose**
Obtain the current position of the scrollbar.

**Synopsis**

```plaintext
function XADscrollbargetpos(id:integer):integer
```

**Argument**

`id` : Scrollbar identifier

**Return value**

Current scrollbar position.

**Related topics**

`XADcreatescrollbar, XADscrollbarset`
XADscrollbarset

Purpose
Set scrollbar characteristics.

Synopsis
procedure XADscrollbarset(id:integer, minimum:integer, maximum:integer,
pages:integer, position:integer)

Arguments
id      Scrollbar identifier
minimum Minimum value for scrollbar
maximum Maximum value for scrollbar
pagesize Size of one "page" (clicking in the scrollbar advances one page at a time)
position Initial position of the scrollbar

Related topics
XADcreatescrollbar, XADscrollbargetpos
XADcreatetree

Purpose
Create a tree object.

Synopsis
procedure XADcreatetree(wid:integer, id:integer, x:integer, y:integer,
w:integer, h:integer)

Arguments
  wid  id of the window containing the tree
  id   Tree identifier
  x    x coordinate (from left of window) of the tree in pixels
  y    y coordinate (from top of window) of the tree in pixels
  w    Width of the tree in pixels
  h    Height of the tree in pixels

Related topics
  XADtreeadd, XADtreereset, XADtreeexpand
**XADtreeadd**

**Purpose**  
Add a branch to a tree

**Synopsis**  
procedure XADtreeadd(id:integer, parent:string, items: set of string)

**Arguments**
- **id**: Tree identifier
- **parent**: Parent branch identifier (single string or comma-separated strings for deeper items)
- **items**: Items to show when parent is expanded

**Example**

XADtreeadd(id_tree,"January",{"New York","London","Paris"})
XADtreeadd(id_tree,"February",{"New York","London","Paris"})
XADtreeadd(id_tree,"January,New York",{"Rainfall","Snowfall"})

**Example result**

```
- January
  - New York
    - Rainfall
    - Snowfall
    - London
    - Paris
  - February
    - New York
    - London
    - Paris
```

**Further information**

Add a branch to the tree. If the **parent** is a single string, it is added as a root item. Then all the **items** are added as its children. If the **parent** is string consisting of comma separated names, XAD will travel down the tree until it finds the corresponding node. Then all the **items** are added as its children.

**Related topics**
- XADcreatetree
- XADtreereset
- XADtreeexpand
XADtreereset

Purpose
Clears the content of a tree

Synopsis
procedure XADtreereset(id:integer)

Argument
id Tree identifier

Related topics
XADcreatetree, XADtreeadd, XADtreeexpand
**XADtreeexpand**

**Purpose**
Expands a tree branch.

**Synopsis**

```pascal
procedure XADtreeexpand(id:integer, parent:string)
```

**Arguments**

- `id`  
  Tree identifier

- `parent`  
  Branch identifier (single string or comma-separated strings for deeper items)

**Related topics**

- `XADcreatetree`
- `XADtreeadd`
- `XADtreereset`
### 6.19 Events specific to objects

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAD_EVENT_CHANGED</td>
<td>Input, editor or scrollbar changed event.</td>
<td>98</td>
</tr>
<tr>
<td>XAD_EVENT_MENU</td>
<td>Menu event.</td>
<td>96</td>
</tr>
<tr>
<td>XAD_EVENT_PRESSED</td>
<td>Button pressed event.</td>
<td>97</td>
</tr>
<tr>
<td>XAD_EVENT_SELECTION</td>
<td>Selection event.</td>
<td>98</td>
</tr>
<tr>
<td>XAD_EVENT_TIMER</td>
<td>Timer event.</td>
<td>96</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_CLOSED</td>
<td>Window closed event.</td>
<td>96</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_CLOSING</td>
<td>Window closing event.</td>
<td>96</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_HIDDEN</td>
<td>Window hidden event.</td>
<td>97</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_MOVED</td>
<td>Window moved event.</td>
<td>97</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_OPENED</td>
<td>Window opened event.</td>
<td>97</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_RESIZED</td>
<td>Window resized event.</td>
<td>97</td>
</tr>
<tr>
<td>XAD_EVENT_WINDOW_SHOWN</td>
<td>Window shown event.</td>
<td>97</td>
</tr>
</tbody>
</table>

**XAD_EVENT_MENU**

**Description**
Menu event.

**Note**
Event generated when a menu item is selected.

**XAD_EVENT_TIMER**

**Description**
Timer event.

**Note**
Timer event associated with *window* objects. See XADwindowsettimer

**XAD_EVENT_WINDOW_CLOSED**

**Description**
Window closed event.

**Note**
Event generated when a *window* was closed using XADwindowclose or by the user.

**XAD_EVENT_WINDOW_CLOSING**

**Description**
Window closing event.

**Note**
Event generated when a *window* is about to be closed. May be overridden by calling XADwindowkeep.
XAD_EVENT_WINDOW_HIDDEN

Description  Window hidden event.
Note  Event generated when a window was hidden using XADwindowhide or by the user.

XAD_EVENT_WINDOW_MOVED

Description  Window moved event.
Note  Event generated when the window position has changed.

XAD_EVENT_WINDOW_OPENED

Description  Window opened event.
Note  Crucial event in the lifetime of a window opened using XADwindowopen. Perform all object initializations when this event is received.

XAD_EVENT_WINDOW_RESIZED

Description  Window resized event.
Note  Event generated when a window is resized by the user. Capture this event in order to update the positions and sizes of objects in the window, using XADsetpos.

XAD_EVENT_WINDOW_SHOWN

Description  Window shown event.
Note  Event generated when a window is shown as a result of calling XADwindowshow. Perform all object initializations when this event is received.

XAD_EVENT_PRESSED

Description  Button pressed event.
Note  Event generated by regular button objects, check buttons, or radio buttons.
**XAD_EVENT_CHANGED**

**Description**  
Input, editor or scrollbar changed event.

**Note**  
Event indicating that the text in an *input* or *editor* object has changed, or that the position of a *scrollbar* has changed.

**XAD_EVENT_SELECTION**

**Description**  
Selection event.

**Note**  
Event indicating that the selection has changed in a *list*, *droplist*, or *tab* object.
### Chapter 7

#### Generic routines

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XADdestroy</td>
<td>Delete an object.</td>
<td>100</td>
</tr>
<tr>
<td>XADenable</td>
<td>Enable/disable user interaction.</td>
<td>101</td>
</tr>
<tr>
<td>XADgeteventtext</td>
<td>Retrieve the message associated with an event.</td>
<td>113</td>
</tr>
<tr>
<td>XADgeth</td>
<td>Get the height of an object.</td>
<td>107</td>
</tr>
<tr>
<td>XADgetmousex</td>
<td>Get the x coordinate of the mouse cursor relative to an object.</td>
<td>102</td>
</tr>
<tr>
<td>XADgetmousey</td>
<td>Get the y coordinate of the mouse cursor relative to an object.</td>
<td>103</td>
</tr>
<tr>
<td>XADgetw</td>
<td>Get the width of an object.</td>
<td>106</td>
</tr>
<tr>
<td>XADgetx</td>
<td>Get the x coordinate of an object.</td>
<td>104</td>
</tr>
<tr>
<td>XADgety</td>
<td>Get the y coordinate of an object.</td>
<td>105</td>
</tr>
<tr>
<td>XADrefresh</td>
<td>Refresh an object.</td>
<td>108</td>
</tr>
<tr>
<td>XADsetfocus</td>
<td>Focus on an object</td>
<td>109</td>
</tr>
<tr>
<td>XADsetpos</td>
<td>Reposition an object.</td>
<td>110</td>
</tr>
<tr>
<td>XADsettext</td>
<td>(Re)set the textual information of an object.</td>
<td>111</td>
</tr>
<tr>
<td>XADsetvisible</td>
<td>Making an object visible/hidden.</td>
<td>112</td>
</tr>
</tbody>
</table>
XADdestroy

Purpose
Delete an object.

Synopsis
procedure XADdestroy(id:integer)

Argument
id
Object identifier

Further information
When an object is no longer needed, it can be destroyed. Once destroyed, its id can no longer be used. However, a new object can be created with the same id.
XADenable

Purpose
Enable/disable user interaction.

Synopsis
procedure XADenable(id:integer, state:boolean)

Arguments
id Object identifier
state true enable user interaction with the object
      false disable user interaction with the object

Further information
Used to enable/disable user interaction with the object. For example, a Save button should probably be disabled if there is nothing to save.
**XADgetmousex**

**Purpose**
Get the x coordinate of the mouse cursor relative to an object.

**Synopsis**
```plaintext
function XADgetmousex(id:integer):integer
```

**Argument**
- **id** Object identifier

**Return value**
- x coordinate of the mouse cursor, relative to the object.

**Further information**
This function returns the x coordinate (in pixels) of the mouse cursor, relative to the upper left corner of the object specified.

**Related topics**
- **XADgetmousey**
**XADgetmousey**

**Purpose**
Get the y coordinate of the mouse cursor relative to an object.

**Synopsis**
```
function XADgetmousey(id:integer):integer
```

**Argument**
id
Object identifier

**Return value**
y coordinate of the mouse cursor, relative to the object.

**Further information**
This function returns the y coordinate (in pixels) of the mouse cursor, relative to the upper left corner of the object specified.

**Related topics**
XADgetmousex
XADgetx

Purpose
Get the x coordinate of an object.

Synopsis
function XADgetx(id:integer):integer

Argument
id  Object identifier

Return value
x coordinate value of the object.

Further information
This function returns the x coordinate (in pixels) of the object id relative to the upper left corner of the window containing it. If id refers to a window, it returns the coordinate relative to the screen.

Related topics
XADgety, XADgeth, XADgetw
**XADgety**

**Purpose**
Get the y coordinate of an object.

**Synopsis**

```plaintext
function XADgety(id:integer):integer
```

**Argument**

- **id**
  Object identifier

**Return value**

- y coordinate value of the object.

**Further information**
This function returns the y coordinate (in pixels) of the object `id` relative to the upper left corner of the window containing it. If `id` refers to a window, it returns the coordinate relative to the screen.

**Related topics**

- `XADgetx`
- `XADgeth`
- `XADgetw`
XADgetw

Purpose
Get the width of an object.

Synopsis
function XADgetw(id:integer):integer

Argument
id Object identifier

Return value
Width of the object.

Further information
This function returns the width (in pixels) of the object. If the identifier \textit{id} represents a window, and its value is negative (e.g. -1000000000), then the \textit{inner} width of the window is returned. This represents the ‘useable’ space of a window, as it excludes borders, menus, titles, etc.

Related topics
XADgeth, XADgetx, XADgety
XADgeth

Purpose
Get the height of an object.

Synopsis
function XADgeth(id:integer):integer

Argument
id Object identifier

Return value
Height of the object.

Further information
This function returns the height (in pixels) of the object. If the identifier id represents a window, and its value is negative (e.g. -1000000000), then the inner height of the window is returned. This represents the ‘useable’ space of a window, as it excludes borders, menus, titles, etc.

Related topics
XADgetw, XADgetx, XADgety
XADrefresh

Purpose
Refresh an object.

Synopsis
procedure XADrefresh(id:integer)

Argument
id Object identifier

Further information
If an object may not be painted correctly due to a complex layout operation, use this routine to update its appearance.
XADsetfocus

Purpose
Focus on an object

Synopsis
procedure XADsetfocus(id:integer)

Argument
id Object identifier

Further information
The target object will receive the keyboard focus.
**XADsetpos**

**Purpose**
Reposition an object.

**Synopsis**

```plaintext
procedure XADsetpos(id:integer, x:integer, y:integer, w:integer, h:integer)
```

**Arguments**

- `id` Object identifier
- `x` X coordinate
- `y` W coordinate
- `w` Width
- `h` Height

**Further information**
This procedure repositions the object `id`. Note: use the `XADget*` routines to leave parameters unchanged.

**Related topics**

- `XADgeth`, `XADgetw`, `XADgetx`, `XADgety`
**XADsettext**

**Purpose**
(Re)set the textual information of an object.

**Synopsis**

procedure XADsettext(id:integer, text:string)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Object identifier</td>
</tr>
<tr>
<td>text</td>
<td>New textual information</td>
</tr>
</tbody>
</table>

**Further information**
This procedure updates the textual information of an object if applicable.
XADsetvisible

Purpose
Making an object visible/hidden.

Synopsis
procedure XADsetvisible(id:integer, visible:boolean)

Arguments
<table>
<thead>
<tr>
<th>id</th>
<th>Object identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>visible</td>
<td>Display option.</td>
</tr>
<tr>
<td>true</td>
<td>object visible</td>
</tr>
<tr>
<td>false</td>
<td>object hidden</td>
</tr>
</tbody>
</table>

Further information
Objects may be shown or hidden by calling this procedure.
XADgeteventtext

Purpose
Retrieve the message associated with an event.

Synopsis
function XADgeteventtext:string

Return value
The event message or an empty string.

Further information
If an event carries textual information (for example a tab selection), use this function to retrieve the text when handling the event.

Related topics
XAD_EVENT_CHANGED, XAD_EVENT_SELECTION, XAD_EVENT_KEYDOWN, XAD_EVENT_KEYUP
Chapter 8
Generic events

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAD_EVENT_KEYDOWN</td>
<td>Key pressed.</td>
<td>p. 114</td>
</tr>
<tr>
<td>XAD_EVENT_KEYUP</td>
<td>Key released.</td>
<td>p. 115</td>
</tr>
<tr>
<td>XAD_EVENT_MOUSE_LEFTDOWN</td>
<td>Left mouse button pressed.</td>
<td>p. 115</td>
</tr>
<tr>
<td>XAD_EVENT_MOUSE_LEFTUP</td>
<td>Left mouse button released.</td>
<td>p. 115</td>
</tr>
<tr>
<td>XAD_EVENT_MOUSE_MOVED</td>
<td>Mouse moved.</td>
<td>p. 115</td>
</tr>
<tr>
<td>XAD_EVENT_MOUSE_RIGHTDOWN</td>
<td>Right mouse button pressed.</td>
<td>p. 115</td>
</tr>
<tr>
<td>XAD_EVENT_MOUSE_RIGHTUP</td>
<td>Right mouse button released.</td>
<td>p. 115</td>
</tr>
</tbody>
</table>

**XAD_EVENT_KEYDOWN**

<table>
<thead>
<tr>
<th>Description</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key pressed.</td>
<td>Indicates that a key was <em>pressed</em> when the object had the focus. Call XADgeteventtext to obtain the representation of the key that was pressed. Letters and digits are returned as themselves. The following special codes can also be returned:</td>
</tr>
<tr>
<td>Up arrow</td>
<td>&quot;up&quot;</td>
</tr>
<tr>
<td>Down arrow</td>
<td>&quot;down&quot;</td>
</tr>
<tr>
<td>Left arrow</td>
<td>&quot;left&quot;</td>
</tr>
<tr>
<td>Right arrow</td>
<td>&quot;right&quot;</td>
</tr>
<tr>
<td>Tab</td>
<td>&quot;tab&quot;</td>
</tr>
<tr>
<td>Enter</td>
<td>&quot;enter&quot;</td>
</tr>
<tr>
<td>Shift</td>
<td>&quot;shift&quot;</td>
</tr>
<tr>
<td>Ctrl</td>
<td>&quot;control&quot;</td>
</tr>
<tr>
<td>Caps Lock</td>
<td>&quot;capslock&quot;</td>
</tr>
<tr>
<td>Esc</td>
<td>&quot;esc&quot;</td>
</tr>
<tr>
<td>Spacebar</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>Page Up</td>
<td>&quot;pageup&quot;</td>
</tr>
<tr>
<td>Page Down</td>
<td>&quot;pagedown&quot;</td>
</tr>
<tr>
<td>End</td>
<td>&quot;end&quot;</td>
</tr>
<tr>
<td>Home</td>
<td>&quot;home&quot;</td>
</tr>
<tr>
<td>Insert</td>
<td>&quot;ins&quot;</td>
</tr>
<tr>
<td>Delete</td>
<td>&quot;del&quot;</td>
</tr>
</tbody>
</table>
### XAD_EVENT_KEYUP

**Description**  
Key released.

**Note**  
Indicates that a key was released when the object had the focus. See XAD_EVENT_KEYDOWN for more information.

---

### XAD_EVENT_MOUSE_LEFTDOWN

**Description**  
Left mouse button pressed.

**Note**  
Indicates that the left mouse button was pressed above the object. The coordinates of the mouse relative to ANY object id can be obtained ANYTIME using XADgetmousex and XADgetmousey.

---

### XAD_EVENT_MOUSE_LEFTUP

**Description**  
Left mouse button released.

**Note**  
Indicates that the left mouse button was released above the object.

---

### XAD_EVENT_MOUSE_MOVED

**Description**  
Mouse moved.

**Note**  
Indicates that the mouse has moved above the object.

---

### XAD_EVENT_MOUSE_RIGHTDOWN

**Description**  
Right mouse button pressed.

**Note**  
Indicates that the right mouse button was pressed above the object.

---

### XAD_EVENT_MOUSE_RIGHTUP

**Description**  
Right mouse button released.

**Note**  
Indicates that the right mouse button was released above the object.
Chapter 9
Utility routines

<table>
<thead>
<tr>
<th>Routine</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XADchoosefile</td>
<td>Display file selection dialog.</td>
<td>121</td>
</tr>
<tr>
<td>XADhandleevents</td>
<td>Handling events during program execution.</td>
<td>120</td>
</tr>
<tr>
<td>XADid</td>
<td>Obtain a unique identifier to represent objects.</td>
<td>117</td>
</tr>
<tr>
<td>XADpopupmenu</td>
<td>Creation of a pop-up menu.</td>
<td>122</td>
</tr>
<tr>
<td>XADsavescreenshot</td>
<td>Take a screenshot of an object and save it to a file.</td>
<td>118</td>
</tr>
<tr>
<td>XADseteventcallback</td>
<td>Set the event handler callback</td>
<td>119</td>
</tr>
</tbody>
</table>
**XADid**

**Purpose**
Obtain a unique identifier to represent objects.

**Synopsis**

```plaintext
function XADid:integer
```

**Return value**
Automatically incremented unique identifier.

**Example**

```plaintext
declarations
  id_win=XADid
  id_canvas=XADid
  id_button=XADid
  ...
end-declarations
```
XADsavescreenshot

Purpose
Take a screenshot of an object and save it to a file.

Synopsis
procedure XADsavescreenshot(id:integer, filename:string)

Arguments
id  Object identifier (may be a window or any other object type)
filename Name of image file (must have one of these extensions: .jpg, .gif, .bmp, .png)

Example

XADsavescreenshot(id_canvas,"canvas.png")
XADsavescreenshot(id_win,"MyXADApplication.jpg")
**XADseteventcallback**

**Purpose**  
Set the event handler callback

**Synopsis**  
procedure XADseteventcallback(handlername:string)

**Argument**  
handlername  Event handler callback

**Further information**  
This procedure registers the procedure *handlername* to act as an event handler callback. All events will be reported through this callback. The callback procedure has this signature:

procedure guievents(id:integer, event:integer)

and it gets called for every possible event. The user has the option to ignore or deal with events through the event handler callback procedure.
XADhandleevents

**Purpose**
Handling events during program execution.

**Synopsis**
procedure XADhandleevents

**Further information**
If a long calculation (e.g. optimization) is initiated by the event handler callback, the user interface will freeze. This is due to the fact that the processor intensive operations take place on the same thread as the code that draws the user interface or responds to user events (after all, we are dealing with an event). To avoid this phenomenon, call XADhandleevents at regular intervals (e.g. during Optimizer callbacks) to allow the user to interact with the user interface. Use caution, however as the following call sequence is likely to occur:

```plaintext
guievents(1,1)
calls   minimize(objective)
calls   globallog
calls   XADhandleevents
calls   guievents(1,10)
calls   ???
```

While an optimization is running events should be dealt with quickly and with little (if well understood) or no side effects. In this situation one could use XADseteventcallback to switch to an alternative, simplified event handler callback.

**Related topics**
XADseteventcallback
XADchoosefile

Purpose
Display file selection dialog.

Synopsis
function XADchoosefile(openorsave:boolean):string
function XADchoosefile(openorsave:boolean, filetypes:string):string

Arguments
openorsave   Dialog type selection.
  true       create an Open file dialog
  false      create a Save file dialog
filetypes   File filters based on file extensions. See example below:

Return value
File name if selection was successful, otherwise an empty string.

Example

  filename:=XADchoosefile(true,"My own type of files (*.myotf)|*.myotf”+
  "|"+
  "MPS files(*.mps)|*.mps"+
  "|"+
  "All Files (*.*)|*.*”+
  "|"

Further information
This is a convenience routine for displaying the standard Windows file selection dialog. Pass true as an argument to create an Open file dialog and false to create a Save file dialog. If the file selection was successful, the returned string contains the file name. If not, the function returns an empty string. The second form of the function allows use of filters based on the file extension.
XADpopupmenu

Purpose
Creation of a pop-up menu.

Synopsis
function XADpopupmenu(menuitems:set of string):string

Argument
menuitems  Menu items

Return value
Selected item, or empty string if no selection was made.

Example
To create a menu with the items Action A and Action B separated by a line, use the following:

    choice:=XADpopupmenu(\"ActionA\", \"XADseparator\", \"Action B\")

Further information
It is customary for user interfaces to display a menu of options when the user right-clicks on something. This routine achieves just that. When the user makes a choice, it is returned as a string. If the user does not select anything, the returned string will be empty.
Chapter 10
Example

A simple assignment problem will be used to illustrate how XAD works with Mosel to create interactive mathematical programming models.

Here is a screenshot of the application:

![Figure 10.1: Interactive mathematical programming model]

The user can modify the preferences by changing the numbers in the input boxes in the upper left corner. After pressing Solve, the results are shown in the diagram. This procedure can be repeated any number of times.

Let's examine the code, piece by piece:

```mosel
model Assignment
uses "mmxprs", "mmxad"

declarations
NP = 5
PF = 1..NP
PREF: array[PF,PF] of integer
assign: array[PF,PF] of integer
end-declarations
```

Here, we tell Mosel that we need to use Xpress-Optimizer (mmxprs) as well as XAD (mmxad) to build this application. We also declare some identifiers to be used later in the model, such as the decision variable array assign.

In the next section:
We assign some suggested preferences (these will be editable by the user). The objective function Satisfaction reflects the overall sum of preferences depending on whether assignments are made or not. Two sets of constraints, one person per project and one project per person complete the mathematical formulation of this model.

Let’s examine the user interface code:

```
23 declarations
24 id_win; id_textpref=2; id_buttonsolve=2;
25 id_canvas=4; id_buttonexit=5; id_inputs=10; id_texts=100;
26 end-declarations

27 XADCreateWindow (id_win,100,100,600,400, "assignment")
28 XADCreateText (id_win, id_textpref, 24, 6, 15, "Preferences")
29 forall (i in 1..NP) do
30 XADCreateInput (id_win, id_inputs+10*i, 24, 14, 1, 2, "PREF[i,j]")
31 end-do
32 XADCreateButton (id_win, id_buttonsolve, 60, 150, 80, 24, "Solve")
33 forall (i in 1..NP) do
34 XADCreateText (id_win, id_texts+[i,i]+1, 100, 5, 15, "PREF[i,j]")
35 end-do
36 XADCreateCanvas (id_win, id_canvas, 250, 20, 300, 300)
37 XADCreateButton (id_win, id_buttonexit, 250, 340, 80, 24, "Exit")
```

In the code above, a new group of declarations is used to assign unique ids to various GUI objects. The window is created first, then the XAD objects are created one by one, with id_win as their parent. Note that id_inputs and id_texts are special in the sense that they are used in combination with i and j to create unique ids for more than one item. Also note how the expression ""+PREF[i,j]" actually fills each input object with the corresponding preference rating.

The use of integers as ids for XAD objects facilitates grouping objects in easy to understand and manage categories (such as all the inputs above). The user should take advantage of this feature, especially for large models/applications.

The canvas id_canvas displays the results of our optimization problem. An entire procedure is dedicated to updating this object with the most recent information:

```
41 procedure UpdateCanvas
42 XADcanvaserase (id_canvas, XAD_WHITE)
43 idrawn the assignment table
44 forall (i in 1..NP) do
45 XADcanvadrawline (id_canvas, [i-1]*60, 0, [i-1]*60, 300, XAD_BLACK)
46 XADcanvadrawline (id_canvas, 0, [i-1]*60, [i-1]*60, 300, XAD_BLACK)
47 forall (j in 1..NP) do
48 if getbool(assign[i,j])<>0 then
49 XADcanvadrawtext (id_canvas, [j-1]*60+5, 1, 1, "*60+5, 1", "*60+5, 1", "*60+5, 1", "*60+5, 1")
50 else
51 XADcanvadrawtext (id_canvas, [j-1]*60+5, 1, 1, "*60+5, 1", "*60+5, 1", "*60+5, 1", "*60+5, 1")
52 end-if
53 end-do
54 end-do
55 XADcanvasrefresh (id_canvas)
56 end-procedure
```

As a general rule, a canvas should be erased first using XADcanvaserase. After all the drawing
is complete, call XADcanvasrefresh to update the contents of the canvas.

The procedure UpdateCanvas draws a grid and then updates each cell based on the optimized values in the assign array. If the assignment is made, a check mark is also drawn in the cell from a bitmap image file. If an assignment is not made, the text in the cell is drawn with a lighter shade of gray to de-emphasize it.

We shall now examine the event handling callback procedure and the code that kick starts the application.

```mosel
procedure guiEvents(id:integer, event:integer)
  if id=id_buttonssolve and event=XAD_EVENT_PRESSED then
    update preferences
    for i in 1..NP do
      PREFERENCES[1,i]=Integer(XADinputgettext(id_inputss[1,ID*i+1])]
    end-do
    update objective function
    Satisfaction=max(m,p in RP) PREFERENCES(m,p)*assign(m,p)
    maximize(Satisfaction)
    update canvas
    UpdateCanvas
  elsif id=id_buttonexit and event=XAD_EVENT_PRESSED then
    XADwindowclose(id_win)
  elsif id=id_via and event=XAD_EVENT_WINDOW_OPENED then
    UpdateCanvas
  end-if
end-procedure
```

Three events are of interest to us in this application: When the window opens and when either of the two buttons is pressed.

- **When the id_buttonssolve button is pressed** Mosel must update the array of preferences (taken directly from the input objects), update the objective function based on the new preferences, optimize the problem, and finally update the canvas so that it shows the new set of assignments.

- **When the id_buttonexit button is pressed**, the window is closed immediately.

- **The event XAD_EVENT_WINDOW_OPENED should be handled to update the status of a window before the user has a chance to interact with it. It is the first event in the lifetime of a window.**

Two more statements in the Mosel code are of interest. We must ensure that XADseteventcallback is called before opening the window, so that the window can send its events to it. Finally, XADwindowopen opens the window, giving it control over the execution (through events).

Note that when we call XADwindowclose or when we close the window with the mouse, execution of the Mosel code in fact continues with the statement following XADwindowopen (in this case there’s nothing else to execute, so the application ends). This means that we should always think of windows as mere components of a Mosel application that temporarily gain control of the Mosel execution through the event handler. Mosel is always in charge and can dismiss a window at any time.

(When developing XAD applications in Xpress-IVE, do not use the Stop feature to end the program execution. This could interrupt the Mosel execution during a system call dealing with the user objects, leaving Mosel and IVE in a corrupt state. Always close all XAD windows instead of using Stop.)

For more instances of XAD applications please refer to the set of examples that accompanies XAD.
Index

A
append item, 49, 54
append text, 28, 36
arc, 81
assignment problem, 123

B
browser, 8, 88
create, 87
button, 9
create, 31
button pressed event, 97

C
callback
event handler, 119
canvas, 9
create, 69
draw arc, 81
draw box, 70
draw chord, 82
draw ellipse, 71
draw image, 74
draw line, 76
draw pie, 80
draw point, 77
draw polygon, 79
draw rectangle, 78
draw text, 83
erase, 72
map, 84
save from file, 75
unmap, 85
update, 73
check button, 10
create, 41
retrieve state, 43
set state, 42
chord, 82
close window, 21
color, 86
color constants, 9
create browser, 87
create button, 31
create canvas, 69
create check button, 41
create droplist, 53
create editor, 35
create group, 47
create input, 32
create list, 48
create multilist, 63
create progress bar, 58
create radio button, 44
create scrollbar, 89
create tab, 60
create text, 27
create tree, 92
create window, 19

d
delete, 6, 100, 108
draw arc, 81
draw box, 70
draw chord, 82
draw ellipse, 71
draw image, 74
draw line, 76
draw pie, 80
draw point, 77
draw polygon, 79
draw rectangle, 78
draw text, 83
drop list, 10
droplist
append item, 54
create, 53
retrieve item, 55
select item, 56
show, 57

e
editor, 10
create, 35
retrieve text, 40
editor changed event, 98
elipse, 71
erase canvas, 72
event, 4, 7
button pressed, 97
editor changed, 98
input changed, 98
key pressed, 114
key released, 115
left mouse button pressed, 115
left mouse button released, 115
menu, 96
mouse moved, 115
right mouse button pressed, 115
right mouse button released, 115
scrollbar changed, 98
selection, 98
timer, 96
window closed, 96
window closing, 96
window hidden, 97
window moved, 97
window opened, 97

126 Xpress Application Developer
window resized, 97
window shown, 97
event callback, 4
event handler callback, 119
event handling, 120
event text, 113
F
file
  load, 37
  save to, 38
file selection dialog, 121
focus, 109
G
get selected tab, 61
group
  create, 47
GUI, 4
H
height, 107
hide window, 23
hiding, 112
I
identifier, 117
input, 11
  create, 32
  retrieve text, 34
input changed event, 98
interactive mathematical programming model, 123
K
keep window, 24
key pressed event, 114
key released event, 115
L
left mouse button pressed event, 115
left mouse button released event, 115
line, 76
list, 12
  append item, 49
  create, 48
  multiple, 12
  retrieve item, 50
  select item, 51
  show, 52
load file, 37
M
making visible, 112
map canvas, 84
menu, 26
menu event, 96
mmxad.dso, 3
Mosel language, 2
MOSEL_DOS, 3
mouse
  x coordinate, 102
  y coordinate, 103
mouse moved event, 115
multilist
create, 63
dynamic, 65
set column name, 66
set list item, 67
show, 64
static, 65
multiple lists, 12
O
object
  delete, 6, 100, 108
  focus, 109
  height, 107
  hidden, 112
  reposition, 110
  textual information, 111
  user interaction, 101
  visible, 112
  width, 106
  x coordinate, 104
  y coordinate, 105
object group, 11
object identifier, 4
open window, 20
P
pie, 80
point, 77
polygon, 79
pop-up menu, 122
progress bar, 13
  create, 58
  set state, 59
R
radio button, 13
  create, 44
  retrieve state, 46
  set state, 45
rectangle, 70, 78
replace text, 29, 33, 39
reposition, 110
reset timer, 25
retrieve item, 50, 55
retrieve state, 43, 46
retrieve text, 30, 34, 40
right mouse button pressed event, 115
right mouse button released event, 115
S
save from file, 75
save to file, 38
screenshot, 118
  any object, 118
scrollbar
  create, 89
  initialize, 91
  position, 90
scrollbar changed event, 98
scrollbar position, 90
scrollbar settings, 91
scrolling, 13
select a tab, 62
select item, 51, 56
selection event, 98
set column name, 66
set list item, 67
set progress, 59
set state, 42, 45
show drop list, 57
show list, 52, 64
show window, 22
T
  tab, 14
    create, 60
text, 14
    append, 28, 36
    create, 27
    replace, 29, 33, 39
    retrieve, 30
textual information, 111
timer event, 96
tree, 15, 93
    add branch, 93
    clear, 94
    create, 92
    expand, 95
    reset, 94
U
unique identifier, 117
unmap canvas, 85
update canvas, 73
user interaction, 101
W
width, 106
window, 15
  close, 21
  create, 19
  hide, 23
  keep, 24
  menu, 26
  open, 20
  reset timer, 25
  show, 22
window close event, 96
window closing event, 96
window hidden event, 97
window moved event, 97
window opened event, 97
window resized event, 97
window shown event, 97
X
x coordinate, 102, 104
XAD, 2
  XAD_BLACK, 9
  XAD_BLUE, 9
  XAD_BOTTOM, 83
  XAD_CENTERH, 83
  XAD_CENTERV, 83
  XAD_CYAN, 9
  XAD_DEFAULT, 83
  XAD_EVENT_CHANGED, 98
  XAD_EVENT_KEYDOWN, 114
  XAD_EVENT_KEYUP, 115
  XAD_EVENT_MENU, 96
  XAD_EVENT_MOUSE_LEFTDOWN, 115
  XAD_EVENT_MOUSE_LEFTUP, 115
  XAD_EVENT_MOUSE_MOVED, 115
  XAD_EVENT_MOUSE_RIGHTDOWN, 115
  XAD_EVENT_MOUSE_RIGHTUP, 115
  XAD_EVENT_PRESSED, 97
  XAD_EVENT_SELECTION, 98
  XAD_EVENTTIMER, 96
  XAD_EVENT_WINDOW_CLOSED, 96
  XAD_EVENT_WINDOW_CLOSING, 96
  XAD_EVENT_WINDOW_HIDDEN, 97
  XAD_EVENT_WINDOW_MOVED, 97
  XAD_EVENT_WINDOW_OPENED, 97
  XAD_EVENT_WINDOW_RESIZED, 97
  XAD_EVENT_WINDOW_SHOWN, 97
  XAD_GREEN, 9
  XAD_LEFT, 83
  XAD_MAGENTA, 9
  XAD_ORANGE, 9
  XAD_RED, 9
  XAD_RIGHT, 83
  XAD_TOP, 83
  XAD_WHITE, 9
  XAD_YELLOW, 9
  XADbrowsergoto, 88
  XADcanvasdrawarc, 81
  XADcanvasdrawbox, 70
  XADcanvasdrawchord, 82
  XADcanvasdrawellipse, 71
  XADcanvasdrawimage, 74
  XADcanvasdrawline, 76
  XADcanvasdrawpie, 80
  XADcanvasdrawpolygon, 79
  XADcanvasdrawrectangle, 77
  XADcanvasdrawtext, 83
  XADcanvaserase, 72
  XADcanvasmapping, 84
  XADcanvasrefresh, 73
  XADcanvasisesavedimage, 75
  XADcanvaseu2map, 85
  XADcheckgetstate, 43
  XADchecksetstate, 42
  XADchooselfile, 121
  XADcolore, 86
  XADcreatebrowser, 87
  XADcreatebutton, 31
  XADcreatecanvas, 69
  XADcreatecheck, 41
  XADcreatedroplist, 53
  XADcreateseditor, 35
  XADcreatesgroup, 47
  XADcreateinput, 32
  XADcreatelist, 48
  XADcreatesmultlist, 63
  XADcreateprogress, 58
  XADcreateradio, 44
  XADcreatescrollbar, 89
  XADcreatetab, 60
  XADcreatetext, 27
  XADcreattree, 92
  XADcreatewindow, 19
  XADdestroy, 100
XADdroplistadd, 54
XADdroplistgetsel, 55
XADdroplistselect, 56
XADdroplistshow, 57
XADeditoraddtext, 36
XADeditorgettext, 40
XADeditorload, 37
XADeditorsave, 38
XADeditorsettext, 39
XADenable, 101
XADgeteventtext, 113
XADgeth, 107
XADgetmousex, 102
XADgetmousey, 103
XADgetw, 106
XADgetx, 104
XADgety, 105
XADhandleevents, 120
XADid, 117
XADinputgettext, 34
XADinputsettext, 33
XADlistadd, 49
XADlistgetsel, 50
XADlistselect, 51
XADlistshow, 52
XADmultilistrefresh, 68
XADmultilistsetcolname, 66
XADmultilistsetsize, 65
XADmultilistsettext, 67
XADmultilistshow, 64
XADpopupmenu, 122
XADprogressbar, 59
XADradiogetstate, 46
XADradiosetstate, 45
XADrefresh, 108
XADSavescreenshot, 118
XADscrollbargetpos, 90
XADscrollbarset, 91
XADseteventcallback, 119
XADsetfocus, 109
XADsetpos, 110
XADsettext, 111
XADsetvisible, 112
XADtabgettab, 61
XADtabsettab, 62
XADtextaddtext, 28
XADtextgettext, 30
XADtextsettext, 29
XADtreeadd, 93
XADtreeexpand, 95
XADtreerest, 94
XADwindowaddmenu, 26
XADwindowclose, 6, 21
XADwindowhide, 6, 23
XADwindowkeep, 24
XADwindowopen, 6, 20
XADwindowsettimer, 25
XADwindowshow, 6, 22

y coordinate, 103, 105