INTERNATIONAL STANDARD

ISO/IEC 15852

First edition
1999-03-01

Information technology — Programming languages — M Windowing API

Technologies de l'information — Langages de programmation — Interface de programme d'application (API) du langage M, utilisant des fenêtres

Reference number
ISO/IEC 15852:1999(E)

Contents

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Tables

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 15852 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

Information technology — Programming languages — M Windowing API

1 General

1.1 Scope

The M Windowing API (MWAPI) extends the M programming technology with the addition of capabilities for developing and operating graphical user interface (GUI) applications.

For the purposes of this International Standard, an application is defined as a collection of one or more M routines using MWAPI capabilities and a user is a person utilizing such an application.

1.2 Relation to Environment

The MWAPI may, but is not required to, operate in conjunction with an underlying windowing platform. The MWAPI does not provide a detailed specification of look and feel for applications. Rather, if an underlying windowing platform is present, it is intended that the MWAPI adhere to the platform's look and feel to the extent practicable. If no underlying windowing platform is present, the MWAPI implementation determines look and feel characteristics and carries out actions that would otherwise be carried out by the underlying windowing platform.

1.3 Windows, Elements and Choices

The primary mechanism for communicating with a user is a window. Windows provide a general mechanism for displaying text and graphics, typically enclosed by a border.

Associated with windows, elements perform specialized tasks. Elements include gadgets, menus, and timers.

Gadgets are used for specialized input and output. For instance, there are gadgets designed for text input and gadgets that enable users to select from a list of items.

Menus are used to enable users to select actions to be performed.

Timers provide a means for notification of an application when a specified period of time has elapsed.

Menus and certain gadgets contain a list of choices. For instance, a list box has a list of items that a user can select.

An entity is a window, element, or choice.

1.4 Displays

Windows are shown to the user via a display, a logical surface on which windows are placed. Each M process has its own display(s); displays are not shared, although physical devices may be shared.

There is not necessarily a one-to-one correspondence between a display and the hardware device on which it appears. Multiple displays may appear on the same device and may overlap. A single display may span multiple devices.

Each display has associated input devices, typically a keyboard with alphanumeric, function and other keys, and a pointer such as a mouse.

1.5 Events

Applications are made aware of actions taken by users, such as pressing a key or selecting a menu item, or of other occurrences via events. Applications specify the events about which they wish to be notified and the code to execute when an event occurs.

1.6 Attributes

The characteristics of windows, elements, and choices are specified through attributes in the structured system variable ^$WINDOW. The characteristics of displays are specified through attributes in the structured system variable ^$DISPLAY. The characteristics of events that occur are specified through attributes in the structured system variable ^$EVENT.

1.7 Drawing

Within one type of gadget, the generic box, the MWAPI provides facilities for drawing geometric figures, text, and other graphical objects.

1.8 Parents and Children

Each window and element has a single parent. A window's parent is either the display on which it appears or another window. An element's parent is a window. If P is the parent of C, then C is a child of P.

A is an ancestor of C and C is a descendant of A if

(i) A is the parent of C, or

(ii) there exist one or more windows (B_1 ... B_n) such that A is the parent of B_1, B_n is the parent of C, and, for x equal to B_1 through B_{n-1}, B_x is the parent of B_{x+1}.

A window or element inherits default values for certain attributes from its parent. A window or element cannot be created before its parent is created. A window or element can be created by the same M command that creates its parent.

P is a parent menu of S and S is a submenu of P if S is displayed when a choice in P is activated.

2 Normative Reference

The following standard contains provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent edition of the standard indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

> ANSI/MDC X11.1-1995, *American National Standard for Information Systems — Programming Languages — M.*

The following terms used in this text have the meaning defined in ANSI/MDC X11.1-1995:

actuallist
command
digit
doargument
eol
expr
expratom
function
glvn
gvn
ident
intexpr
intlit
killargument
L
lvn
mnemonicspace
name
namevalue
numexpr
numlit
openargument
postcond
process
routine
setdestination
SP
ssvn
structured system variable
subscript
svn
timeout
tvexpr
useargument
V
xargument

In addition, the syntax metalanguage used in this International Standard is the same as that used in ANSI/MDC X11.1.

3 Attributes

Attributes are specified as nodes within the structured system variables (ssvns) ^$WINDOW, ^$DISPLAY and ^$EVENT. Attributes are identified via keywords that appear as subscripts within these ssvns. These keywords are defined as dattribute, wattribute, elattribute, cattribute, especattribute, and einfoattribute values, and are referred to collectively as attribute names. All attribute names not specified by this International Standard are reserved.

From the perspective of any M process, ^$DISPLAY, ^$WINDOW and ^$EVENT contain information about only those displays, windows and events that are accessible to the process.

3.1 Manipulating Attributes

When an M process begins execution, ^$WINDOW and ^$EVENT are not defined.

Values may be assigned to attributes by referencing ^$WINDOW, ^$DISPLAY or ^$EVENT as a glvn on the left hand side of the equal sign in an M MERGE command or as a setdestination in an M SET command.

When attribute values are assigned with an M MERGE command, all visible effects occur at the successful conclusion of the command. Note that this does not require that the MERGE command modify an ssvn in an atomic fashion. Rather, it places the visible effects at the conclusion of the command.

If an application attempts to assign a value to a reserved attribute name, an error condition occurs with $ECODE containing ",M?1," and $EREF containing a value that indicates the ssvn reference for which the error occurred.

An entity is created at the conclusion of the first M command that assigns a value to one or more of its attributes. If a value is not explicitly assigned to an attribute when an entity is created, a default value may be provided by the MWAPI or the underlying windowing platform. If the default value is dependent upon the values of one or more attributes of another existing entity, the attribute values of that other entity at the time of the new entity's creation are used. Subsequent changes to the existing entity's attribute values do not affect previously established default values.

> Note: If there are multiple attributes for which values must be assigned when an entity is created, the entity can only be created using the M MERGE command. All gadgets currently defined by this International Standard have this characteristic.

Certain attributes of an existing display, window, element, choice or event information structure can be modified by assigning new values to them. Attribute values are also modified by the MWAPI in response to user actions or other occurrences.

If a value domain is specified for an attribute, all values outside the specified domain are reserved. Unless specifically stated otherwise, if an application attempts to assign to an attribute a value that is outside the specified domain of that attribute, an error condition occurs with $ECODE containing ",M?2," and $EREF containing a value that indicates the ssvn reference for which the error occurred.

The value of an attribute, whether explicitly assigned by the application, or present by default, or set by the MWAPI, can be obtained by referencing ^$WINDOW, ^$DISPLAY or ^$EVENT.

In any context in which an attribute is modifiable, using the M KILL command to remove the value of an attribute causes the attribute to have the default value that it would have if the entity were created at that instant, if a default is specified, or to become undefined, if no default is specified.

In any context in which an attribute is not modifiable, an attempt by an application to assign a value to the attribute or to remove the value of the attribute is ignored.

An MTERM window that was created by the execution of an OPEN command is destroyed by using its wname as the device designation in an M CLOSE command. Any other window, an element, or a choice is destroyed by removing its wname, ename, or item, respectively, from ^$WINDOW with the M KILL command. Destroying an entity causes it to become undefined and causes all of the attributes of the entity to be removed from ^$WINDOW. When a window is destroyed, all descendant windows and elements are destroyed. When an element is destroyed, all descendant choices are destroyed

Execution of an M KILL command which has a killargument equal to "^$WINDOW" or "^$W" causes all windows defined in ^$WINDOW to be destroyed and causes the ^$WINDOW ssvn to become undefined. When a process executes an M HALT command, all of the windows that belong exclusively to it are destroyed and ^$EVENT becomes undefined.

3.2 Attribute Definitions

Within this document, each attribute is defined with a table of the following format:

Definition — Specifies the meaning of the attribute.

Domain — Specifies the range of possible values for the attribute.

Access — Specifies the ways in which the attribute can be accessed by an application:

Create — Indicates that the application can assign a value for the attribute when the entity is created.

Modify — For a wattribute, elattribute, cattribute, or especattribute, Modify indicates that the application can assign a value to the attribute after the entity is created. For a dattribute or einfoattribute, Modify indicates that the application can assign a value to the attribute.

Reference — Indicates that the application can obtain the current value of the attribute.

Default — For a wattribute, elattribute, cattribute, or especattribute, Default specifies the value of the attribute if no value is assigned when the entity is created. For a dattribute or einfoattribute, Default specifies the value of the attribute prior to any assignment of a value to it by the application. In addition to specific default values, the following general default value types are defined.

None — Indicates that no default value is defined and the attribute node is undefined.

Platform — Indicates that the default is established by the underlying windowing platform, if there is one, and by the implementation of the MWAPI otherwise.

Not Applicable — Indicates that no default applies because the attribute can only be referenced.

3.3 Attribute Value Domains

3.3.1 Length of Character Strings

When the MWAPI is present, the portability requirement in Section 2 of the MUMPS International Standard is modified such that any result, whether intermediate or final, of expression evaluation whose length does not exceed the portability limit for longchars, and which would otherwise be permitted, is permitted. This does not permit assignment of a value to an ssvn node whose domain is not longchars, to an lvn, or to a gvn where such assignment would not be permitted by the MUMPS International Standard. This also does not permit use of a value as a subscript, as an actuallist parameter, or as an xargument where such use would not be permitted by the MUMPS International Standard.

3.3.2 Colors

Color specifications have three components: a red value, a green value, and a blue value.

color ::= expr V rcolor , gcolor , bcolor

rcolor ::= intlit

gcolor ::= intlit

bcolor ::= intlit

rcolor specifies the amount of red, gcolor specifies the amount of green, and bcolor specifies the amount of blue. rcolor, gcolor, and bcolor values are restricted to a range of 0, meaning no color, through 65535, meaning full color.

A color value of "0,0,0" signifies black; a color value of "65535,65535,65535" indicates white; and a color value with identical rcolor, gcolor, and bcolor values signifies a shade of grey.

Appropriate scaling from the 0 to 65535 range to the color domain of the underlying windowing platform or hardware is performed automatically by the MWAPI.

3.3.3 Characters and the Empty String

chars denotes a sequence of zero or more characters.

noncommachars denotes a sequence of zero or more characters, excluding the comma character.

longchars denotes a sequence of zero or more characters.

emptystring denotes an M empty string.

tab denotes a tab character.

linend denotes an M eol.

mnemonichar denotes an ampersand character.

3.3.4 True/False Values

truefalse ::= expr

Any truefalse value assigned by an application to an attribute is interpreted as TRUE or FALSE in accordance with the M definition of tvexpr. If the interpretation is TRUE, the attribute's value is set to 1 (one). If the interpretation is FALSE, the attribute's value is set to 0 (zero).

Any truefalse value assigned by the MWAPI has a value of 1 (one) for TRUE and 0 (zero) for FALSE.

3.3.5 Implementation and Application Attributes

implementationattributes, implementationvalues, and implementationdrawcommands are reserved for implementors.

implementationattribute ::= Z unspecified

implementationvalue ::= Z unspecified

implementationdrawcommand ::= Z unspecified

applicationattributes are reserved for application use.

applicationattribute ::= Ychars

An applicationattribute must be a valid M subscript.

The meanings of implementationattributes, implementationvalues, applicationattributes, and implementationdrawcommands are not specified.

3.3.6 External Resources

External resources provided by the implementor, by the underlying windowing platform, or by another source, are identified by extresource.

```
extresource ::= expr V  | F,fileid              |
                        | R,resourceid          |
                        | implementationvalue   |
```

fileid and resourceid are platform-dependent file and resource identifiers, respectively. The formats of fileids and resourceids are unspecified.

In addition to external sources for symbols, a number of symbol types are specified by the MWAPI.