

Information technology — Programming languages — M Windowing API

ICS 35.060

National foreword

This British Standard reproduces verbatim ISO/IEC 15852:1999 and implements it as the UK national standard.

The UK participation in its preparation was entrusted to Technical Committee IST/5, Programming languages, which has the responsibility to:

- aid enquirers to understand the text;
- present to the responsible international/European committee any enquiries on the interpretation, or proposals for change, and keep the UK interests informed;
- monitor related international and European developments and promulgate them in the UK.

A list of organizations represented on this committee can be obtained on request to its secretary.

Cross-references

The British Standards which implement international or European publications referred to in this document may be found in the BSI Standards Catalogue under the section entitled “International Standards Correspondence Index”, or by using the “Find” facility of the BSI Standards Electronic Catalogue.

A British Standard does not purport to include all the necessary provisions of a contract. Users of British Standards are responsible for their correct application.

Compliance with a British Standard does not of itself confer immunity from legal obligations.

Summary of pages

This document comprises a front cover, an inside front cover, the ISO/IEC title page, pages ii to vii, a blank page, pages 1 to 85 and a back cover.

The BSI copyright notice displayed in this document indicates when the document was last issued.

This British Standard, having been prepared under the direction of the DISC Board, was published under the authority of the Standards Committee and comes into effect on 15 September 1999

© BSI 09-1999

ISBN 0 580 32946 1

Amendments issued since publication

Amd. No.	Date	Comments

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*



Contents

Page

1	General	1
1.1	Scope	1
1.2	Relation to Environment	1
1.3	Windows, Elements and Choices	1
1.4	Displays	2
1.5	Events	2
1.6	Attributes	2
1.7	Drawing	2
1.8	Parents and Children	2
2	Normative Reference	3
3	Attributes	4
3.1	Manipulating Attributes	4
3.2	Attribute Definitions	5
3.3	Attribute Value Domains	6
3.3.1	Length of Character Strings	6
3.3.2	Colors	6
3.3.3	Characters and the Empty String	6
3.3.4	True/False Values	7
3.3.5	Implementation and Application Attributes	7
3.3.6	External Resources	7
3.3.7	Fonts	8
3.3.8	Identifiers	9
3.3.9	Position, Size and Units	10
3.3.10	Display Characteristics	11
3.3.11	Drawing Characteristics	12
3.3.12	Orientation	13
3.3.13	Key and Pointer Button Codes	13
3.3.14	Insertion Point and Selection Range	13
3.3.15	Title Position	13
3.3.16	Choice Layout	14
3.3.17	Modality	14
3.3.18	Scrolling	14
3.3.19	Focus Location	15
4	Displays	15
4.1	Display Attributes	15
4.1.1	BCOLOR	16
4.1.2	CLIPBOARD	16

4.1.3	COLOR	16
4.1.4	COLORTYPE	16
4.1.5	FCOLOR	16
4.1.6	FOCUS	17
4.1.7	KEYBOARD	17
4.1.8	PEN	17
4.1.9	PLATFORM	17
4.1.10	PTR	18
4.1.11	SIZE	18
4.1.12	SPECTRUM	18
4.1.13	TYPEFACE	18
4.1.14	UNITS	19
5	Windows	19
5.1	Window Types	19
5.1.1	MTERM Windows	20
5.2	Window Components	20
5.3	Window Repainting	20
5.4	Window Attributes	20
5.4.1	Applicability of Window Attributes	21
5.4.2	ACTIVE	23
5.4.3	BCOLOR	23
5.4.4	COLOR	23
5.4.5	DEFBUTTON	23
5.4.6	DISPLAY	24
5.4.7	EVENT	24
5.4.8	FCOLOR	24
5.4.9	FFACE	24
5.4.10	FSIZE	24
5.4.11	FSTYLE	25
5.4.12	ICON	25
5.4.13	ICONIFY	25
5.4.14	ID	25
5.4.15	ITITLE	25
5.4.16	MENUBAR	26
5.4.17	MIN	26
5.4.18	MODAL	26
5.4.19	NEXTG	26
5.4.20	PARENT	27
5.4.21	POS	27
5.4.22	RESIZE	28
5.4.23	SCROLL	28
5.4.24	SIZE	28
5.4.25	SIZEMIN	29
5.4.26	SIZEWIN	29
5.4.27	TIED	29
5.4.28	TITLE	30
5.4.29	TYPE	30
5.4.30	UNITS	30
5.4.31	VISIBLE	30
6	Elements	31
6.1	Gadgets	31
6.1.1	Check Box	31
6.1.2	Document	32
6.1.3	Generic Box	32
6.1.4	Group Frame	32
6.1.5	Label	32

6.1.6	List Box	32
6.1.7	List Button	32
6.1.8	List Entry Box	32
6.1.9	Long List Box	33
6.1.10	Push Button	33
6.1.11	Radio Button Set	33
6.1.12	Scroll	33
6.1.13	Symbol	33
6.1.14	Text	33
6.2	Menus	33
6.3	Timers	34
6.4	Element Attributes	34
6.4.1	ACTIVE	38
6.4.2	BCOLOR	38
6.4.3	CANCEL	38
6.4.4	CANCHANGE	38
6.4.5	CHANGED	38
6.4.6	CHARMAX	39
6.4.7	CHOICE	39
6.4.8	DRAW	39
6.4.9	DRAWTYPE	40
6.4.10	EVENT	40
6.4.11	FCOLOR	40
6.4.12	FFACE	40
6.4.13	FRAMED	40
6.4.14	FSIZE	41
6.4.15	FSTYLE	41
6.4.16	ID	41
6.4.17	INSELECT	41
6.4.18	INTERVAL	42
6.4.19	NEXTG	42
6.4.20	POS	43
6.4.21	RESOURCE	43
6.4.22	ROWCOL	44
6.4.23	SCROLL	44
6.4.24	SCROLLBY	45
6.4.25	SCROLLDIR	45
6.4.26	SCROLLPOS	45
6.4.27	SCROLLRANGE	46
6.4.28	SELECTMAX	46
6.4.29	SELECTVAL	46
6.4.30	SIZE	47
6.4.31	TBCOLOR	48
6.4.32	TFCOLOR	48
6.4.33	TFFACE	48
6.4.34	TFSIZE	48
6.4.35	TFSTYLE	49
6.4.36	TITLE	49
6.4.37	TOPSHOW	49
6.4.38	TPOS	50
6.4.39	TYPE	50
6.4.40	UNITS	50
6.4.41	VALUE	50
6.4.42	VISIBLE	52
6.5	Choice Attributes	52
6.5.1	Applicability of Choice Attributes	53
6.5.2	ACCELERATOR	53
6.5.3	ACTIVE	53

6.5.4	AID	54
6.5.5	EVENT	54
6.5.6	MARKER	54
6.5.7	SEPARATOR	54
6.5.8	SUBMENU	55
7	Drawing	55
7.1	Draw Command Numbers	55
7.2	Draw Commands	56
7.2.1	ARC	56
7.2.2	BCOLOR	57
7.2.3	BITMAP	57
7.2.4	BOX	57
7.2.5	DMODE	57
7.2.6	DRAWTEXT	58
7.2.7	FCOLOR	58
7.2.8	FFACE	58
7.2.9	FILLPAT	58
7.2.10	FSIZE	59
7.2.11	FSTYLE	59
7.2.12	LINE	59
7.2.13	OVAL	59
7.2.14	PENSIZe	59
7.2.15	PICTURE	60
7.2.16	PIE	60
7.2.17	POINT	60
8	Events	60
8.1	Event Types	60
8.1.1	Window State Events	61
8.1.2	Pointer Events	61
8.1.3	Keyboard Events	62
8.1.4	Focus Events	62
8.1.5	Select Events	63
8.1.6	Long List Box Events	64
8.1.7	Help Events	64
8.1.8	Timer Events	64
8.1.9	Applicability of Event Types	64
8.2	Event Specification Attributes	68
8.2.1	ENABLE	68
8.2.2	FILTERIN	68
8.3	Processing Events	69
8.4	Event Commands	69
8.4.1	ESTART	69
8.4.2	ESTOP	70
8.4.3	ETRIGGER	70
8.5	Event Information	71
8.5.1	CHOICE	71
8.5.2	CLASS	72
8.5.3	ELEMENT	72
8.5.4	KEY	72
8.5.5	NEXTFOCUS	72
8.5.6	PRIORFOCUS	73
8.5.7	OK	73
8.5.8	PBUTTON	73
8.5.9	PPOS	73
8.5.10	PSTATE	74
8.5.11	SEQUENCE	74

8.5.12	TYPE.....	74
8.5.13	WINDOW	74
8.5.14	Applicability of Event Information Attributes.....	75
9	Additional Functions and Special Variables.....	78
9.1	Functions	78
9.1.1	\$WFONT	78
9.1.2	\$WTFIT	78
9.1.3	\$WTWIDTH.....	79
9.2	Special Variables.....	79
10	Error Handling.....	79
10.1	Error Information.....	79
10.2	Errors	80
11	Conformance.....	81
11.1	Implementations	81
11.1.1	Conforming Implementation.....	81
11.1.2	MDC Conforming Implementation	83
11.1.3	MDC Strictly Conforming Implementation.....	83
11.1.4	MDC Partial Implementation	83
11.1.5	Multiple Levels of Conformance.....	83
Index		84

Tables

Window Attributes	22
Element Attributes	36
Choice Attributes	53
Events	65
Event Information	76
Implementation Definition.....	81
Implementation-Specific Features	81
Portability Limits.....	82
Unimplemented Features.....	82

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 ($B_1 \dots B_n$) 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:

actualist
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 infoattribute 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 <u>wattribute</u> , <u>elattribute</u> , <u>cattribute</u> , or <u>especattribute</u> , Modify indicates that the application can assign a value to the attribute after the entity is created. For a <u>dattribute</u> or <u>einfoattribute</u> , 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 <u>wattribute</u> , <u>elattribute</u> , <u>cattribute</u> , or <u>especattribute</u> , Default specifies the value of the attribute if no value is assigned when the entity is created. For a <u>dattribute</u> or <u>einfoattribute</u> , 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 actulist 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 ::= Y chars

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, <u>fileid</u>
R, <u>resourceid</u>
<u>implementationvalue</u>

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.

		<u>extresource</u>	
<u>symbol</u> ::=	<u>expr V</u>	M,ERROR M,INFO M,QUEST M,WARN	

symbol values of "M,ERROR", "M,INFO", "M,QUEST", and "M,WARN" specify symbols that identify error, information, question, and warning situations, respectively. If the underlying windowing platform provides symbols for these situations, they are used. Otherwise, symbols are provided by the MWAPI. extresource enables use of other symbols not specified by the MWAPI.

		<u>extresource</u>	
<u>marker</u> ::=	<u>expr V</u>	M,BULLET M,CHECK M,DIAMOND	

marker values of "M,BULLET", "M,CHECK", and "M,DIAMOND" specify menu choice markers shaped like bullets, check marks, and diamonds, respectively. If the underlying windowing platform provides markers for these situations, they are used. Otherwise, markers are provided by the MWAPI. extresource enables use of other markers not specified by the MWAPI.

3.3.7 Fonts

Type fonts specify the appearance characteristics of text. A font specification consists of three components: a face, a size, and a style.

<u>fontface</u> ::= <u>expr V</u>	M,DEFAULT M,FIXED M,SANS M,SERIF <u>alpha chars</u>
-----------------------------------	---

The following fontface values are defined:

- M,DEFAULT Default type face of the underlying windowing platform.
- M,FIXED A non-proportional type face.
- M,SANS A proportional type face from a sans-serif family.
- M,SERIF A proportional type face from a serif family.

fontface values not beginning with "M," may be used by implementors to identify additional type faces.

fontsize ::= expr V numlit

Positive numeric values for fontsize specify the size of displayed text measured in points. A point is 1/72 inches.

<u>fontstyle</u> ::= <u>expr</u> V		NORMAL	
		<u>fstyle</u> [, <u>fstyle</u>] ...	

<u>fstyle</u> ::=		BOLD	
		ITALIC	
		ULINE	
		<u>implementationvalue</u>	

If fontstyle includes the fstyle BOLD, the text is displayed in bold face. If fontstyle includes the fstyle ITALIC, the text is displayed italicized. If fontstyle includes the fstyle ULINE, the text is displayed underlined.

If a font requested by an application cannot be provided, an alternative font with different fontface, fontsize, and/or fontstyle values may be substituted by the underlying windowing platform or the MWAPI implementation. Any references to attributes identifying fonts (including attributes to which values beginning with "M," have been assigned) yield information about the actual font.

3.3.8 Identifiers

A wident is a sequence of letters and/or numbers optionally preceded by a percent sign.

<u>wident</u> ::=		%		[<u>alphanum</u>] ...
		<u>alphanum</u>		

<u>alphanum</u> ::=		<u>ident</u>	
		<u>digit</u>	

Windows are identified by wnames.

wname ::= wident

Elements are identified by enames.

<u>ename</u> ::=		<u>gname</u>	
		<u>mname</u>	
		<u>tname</u>	

Gadgets are identified by gnames.

gname ::= wident

Menus are identified by mnames.

mname ::= wident

Timers are identified by tnames.

tname ::= wident

Choices are identified by items.

item ::= chars

For any attribute which has as its domain wname or ename, if an application attempts to assign (via a command) a value for the attribute which does not identify a window or element which exists at the conclusion of that command, an error condition occurs with \$ECODE containing ",M?3," and \$EREF containing a value that indicates the ssvn reference to which the application attempted to assign the invalid wname or ename. Subsequent destruction of the window or element does not cause an error condition to occur.

3.3.9 Position, Size and Units

A UNITS dattribute, wattribute, or elattribute specifies the unit of measure for the size or position of a display, window, or element, respectively.

<u>unitspec</u> ::=	CHAR [, <u>chars</u>] PIXEL POINT REL [, <u>hscale</u> , <u>vscale</u>] <u>implementationvalue</u>
---------------------	--

A unitspec value of CHAR specifies that position or size is measured relative to the size of characters in a basis font which is determined by context. A unit in the vertical dimension is equal to the line spacing (including external leading) in that font. If chars is specified and is not equal to emptystring, a unit in the horizontal dimension is equal to the average character width in that font for the characters in chars. If chars is not specified or if chars is equal to emptystring, a unit in the horizontal dimension is equal to the maximum character width in that font.

A unitspec value of PIXEL specifies that position or size is measured in pixels. A unit is equal to the size of a pixel.

A unitspec value of POINT specifies that position or size is measured in points. A unit is equal to the size of a point, 1/72 inches.

A unitspec value of REL specifies a relative scale. If hscale and vscale are not given, they default to 100. REL units are defined relative to a basis size of another object. If the other object is a display, the basis size is the size of the display. If the other object is a window, the basis size is the size of the window viewport when that other window is created.

If hbasis and vbasis are the basis size in PIXEL units, and hrel and vrel are a size or position specification in REL units, hrel and vrel are converted to PIXEL units (hpix and vpix) by the following calculations:

$$\text{hpix} = \text{hrel} / \text{hscale} * \text{hbasis}$$

$$\text{vpix} = \text{vrel} / \text{vscale} * \text{vbasis}$$

hscale ::= numlit

vscale ::= numlit

A position consists of a horizontal location (hpos), a vertical location (vpos), and an optional units of measure designation (unitspec).

pos ::= expr V hpos , vpos [, unitspec]

hpos ::= numlit

vpos ::= numlit

The position of an entity is always specified as the location of the origin of that entity with respect to the origin of a display, window or gadget. The origin of a display or a window frame or a window viewport or a gadget is its upper left hand corner. When a window is created, the origin of its application area is the upper left hand corner of the window viewport. Subsequent scrolling moves the application area origin. The origin is position (0,0) and positions are measured in a positive direction to the right and down.

Note: In the future, support may be added for different coordinate systems to facilitate support of languages with orientations other than left-to-right and top-to-bottom.

A size consists of a horizontal size (hsize), a vertical size (vsize), and an optional units of measure designation (unitspec).

<u>size</u> ::= <u>expr</u> <u>V</u>	<u>hsize</u> [, [<u>vsize</u>] [, <u>unitspec</u>]] , <u>vsize</u> [, <u>unitspec</u>]
--------------------------------------	---

hsize ::= numlit

vsize ::= numlit

Unless indicated otherwise, if an application attempts to assign to an attribute an hsize or vsize value which is less than or equal to zero, an error occurs with \$ECODE containing "M?2," and \$EREF containing a value that indicates the ssvn reference for which the error occurred.

If the unitspec is omitted when a pos or size value is assigned to an attribute, unitspec defaults to the then current value of the entity's UNITS attribute.

Any reference to ^\$DISPLAY or ^\$WINDOW for a pos or size value yields a value that includes hpos, vpos, and unitspec pieces and that is expressed in the unit of measure specified (when the reference is made) by the entity's UNITS attribute.

If a size or position value assigned by an application cannot be provided, a size or position is determined by the underlying windowing platform or the MWAPI implementation. Any reference to attributes identifying a size or position reflects the actual size or position.

3.3.10 Display Characteristics

displaycolor specifies the capability of the display to show colors or shades of grey.

<u>displaycolor</u> ::= <u>expr</u> <u>V</u>	COLOR GREY MONO <u>implementationvalue</u>
--	---

A displaycolor value equal to "COLOR" indicates a display that can show colors. A displaycolor value equal to "GREY" indicates a display that can show multiple shades of grey. A displaycolor value equal to "MONO" indicates a display that can show only a single background and foreground color.

platformversion identifies the name and version designator of the underlying windowing platform.

platformversion ::= expr V [platformname , versionid]

<u>platformname</u> ::=	MAC MSWIN PM X/MTF X/OPNLK <u>implementationvalue</u>
-------------------------	--

A platformname value of "MAC" indicates an Apple Macintosh system. A platformname value of "MSWIN" indicates a Microsoft Windows system. A platformname value of "PM" indicates a Presentation Manager system. A platformname value of "X/MTF" indicates an XWindow/Motif system. A platformname value of "X/OPNLK" indicates an XWindow/Open Look system.

versionid ::= chars

3.3.11 Drawing Characteristics

drawtype specifies the type of drawcommands that can be specified.

<u>drawtype</u> ::= <u>expr V</u>	MDRAW <u>implementationvalue</u>
-----------------------------------	-------------------------------------

fillmode specifies the relationship between components of a drawing and the draw background color.

<u>fillmode</u> ::= <u>expr V</u>	OPAQUE TRANSPARENT <u>implementationvalue</u>
-----------------------------------	---

fillpattern specifies the pattern used to fill an area.

<u>fillpattern</u> ::= <u>expr V</u>	BDIAGONAL CROSSHATCH DCROSSHATCH FDIAGONAL HORIZONTAL NONE SOLID VERTICAL <u>implementationvalue</u>
--------------------------------------	--

dmode specifies the effect of drawing by certain drawcommands on existing pixels in the drawing area.

<u>dmode</u> ::= <u>expr V</u>	SOLID XOR <u>implementationvalue</u>
--------------------------------	--

3.3.12 Orientation

An orientation value equal to "H" indicates that an entity has horizontal orientation. An orientation value equal to "V" indicates that an entity has vertical orientation.

<u>orientation</u> ::= <u>expr</u> V		H	
		V	
		<u>implementationvalue</u>	

3.3.13 Key and Pointer Button Codes

Key sequences, consisting of one or more keys, are identified by key codes (keycodes).

keycode ::= noncommachars

Pointer buttons are identified by pointer button codes (pbuttoncodes).

pbuttoncode ::= noncommachars

The correspondence between actual keyboard keys and key codes and between actual pointer buttons and pointer button codes is established by a mapping mechanism. For each key code, the mapping mechanism specifies the corresponding key sequence and whether the key code is a data key code or a function key code. For each pointer button code, the mapping mechanism specifies the corresponding pointer button.

While this International Standard does not specify how this mapping is carried out, a mapping mechanism must be available.

3.3.14 Insertion Point and Selection Range

insel ::= expr V insert , seloff

insert ::= intlitt

seloff ::= [-] intlitt

insert indicates a position within a text value. A value equal to 0 (zero) indicates a position before the first character. A value of N, where N>0, indicates a position after the Nth character and, if the N+1th character is present, before the N+1th character.

seloff indicates an offset relative to insert.

3.3.15 Title Position

titlepos specifies where a gadget title appears.

<u>titlepos</u> ::= <u>expr</u> V		LEFT	
		RIGHT	
		TOP	
		<u>implementationattribute</u>	

If titlepos is "LEFT", the upper right hand corner of the title is immediately adjacent to the upper left hand corner of the gadget. If titlepos is "RIGHT", the upper left hand corner of the title is immediately adjacent to the upper right hand corner of the gadget. If titlepos is "TOP", the lower left hand corner of the title is immediately adjacent to the upper left hand corner of the gadget.

3.3.16 Choice Layout

rowcolspec specifies how choices are positioned within a gadget.

rowcolspec ::= expr V rowcol [, rowcolsize [, filldir]]

rowcol ::=

COL
ROW

rowcolsize ::= intlitt

filldir ::=

H
V

The meaning of rowcolspec is specified in Section 0

3.3.17 Modality

modalitytype specifies the extent to which a window with the MODAL wattribute defined disables other windows.

modalitytype ::= expr V

ANCESTORS
APPLICATION
PARENT
<u>implementationvalue</u>

The meaning of modalitytype is specified in Section 0.

3.3.18 Scrolling

scrollrange specifies the value range that a scroll or long list box gadget displays.

scrollrange ::= expr V

<u>scrollmin</u>
[<u>scrollmin</u>] , <u>scrollmax</u>

scrollmin ::= numlitt

scrollmax ::= numlitt

scrollby specifies the amount by which the VALUE elattribute increases or decreases when the user presses a scroll gadget's movement controls.

scrollby ::= expr V

<u>scrollsmall</u>
[<u>scrollsmall</u>] , <u>scrollbig</u>

scrollsmall ::= numlitt

scrollbig ::= numlitt

3.3.19 Focus Location

focusloc identifies a window or gadget which has, had, or is about to get focus.

<u>focusloc</u> ::= <u>expr</u> V	<u>wname</u> <u>wname</u> , <u>gname</u> <u>emptystring</u>
-----------------------------------	---

If focusloc identifies a window, it has the form:

wname

If focusloc identifies a gadget, it has the form:

wname , gname

If focusloc does not identify a window or gadget defined in ^\$WINDOW, it has the form:

emptystring

4 Displays

The ssvn ^\$DISPLAY contains information about the displays accessible to a process, in the form

^\$DI[SPLAY] (display , dattribute)

display is the identifier for a display.

display ::= expr

The format of display values and the means for associating displays with actual hardware devices are not specified except that the display value must be a valid subscript.

4.1 Display Attributes

Display attributes, identified by dattributes, define the characteristics of displays.

<u>dattribute</u> ::= <u>expr</u> V	BCOLOR CLIPBOARD COLOR COLORTYPE FCOLOR FOCUS KEYBOARD PEN PLATFORM PTR SIZE SPECTRUM TYPEFACE UNITS <u>applicationattribute</u> <u>implementationattribute</u>
-------------------------------------	--

4.1.1 BCOLOR dattribute

Definition	BCOLOR indicates the default value for the BCOLOR <u>wattribute</u> for windows that are children of the display.
Domain	<u>color</u>
Access	Modify, Reference
Default	Platform

4.1.2 CLIPBOARD dattribute

Definition	CLIPBOARD specifies the data associated with the clipboard provided by the underlying windowing platform, if there is one, and by the MWAPI implementation, otherwise.
Domain	<u>longchars</u>
Access	Modify, Reference
Default	Not Applicable

4.1.3 COLOR dattribute

Definition	COLOR indicates the default value for the COLOR <u>wattribute</u> for windows that are children of the display.
Domain	<u>color</u>
Access	Modify, Reference
Default	Platform

4.1.4 COLORTYPE dattribute

Definition	COLORTYPE indicates the color capabilities of the display.
Domain	<u>displaycolor</u>
Access	Reference
Default	Not Applicable

4.1.5 FCOLOR dattribute

Definition	FCOLOR indicates the default value for the FCOLOR <u>wattribute</u> for windows that are children of the display.
Domain	<u>color</u>
Access	Modify, Reference
Default	Platform

4.1.6 FOCUS dattribute

Definition FOCUS identifies the window and gadget, if any, which has focus; that is, to which keyboard and pointer input associated with this display are sent. If no window created by the application has focus, or if the window or gadget which has focus is destroyed, the value is platform dependent.

If an application assigns a value to FOCUS which identifies a window or gadget defined in ^\$WINDOW, that window or gadget receives focus but a focus event does not occur.

If an application attempts to set focus to a window or gadget that is not currently visible and active, or to set focus to a gadget that cannot get focus in the underlying windowing platform, an error condition occurs with \$ECODE containing ",M?4," and \$EREF containing a value that indicates the ssvn reference for which the error occurred.

Domain focusloc

Access Modify, Reference

Default Platform

4.1.7 KEYBOARD dattribute

Definition If KEYBOARD is defined, the display has a keyboard. If KEYBOARD is not defined, the display does not have a keyboard.

Domain emptystring

Access Reference

Default Not applicable

4.1.8 PEN dattribute

Definition If PEN is defined, the display has a pen. If PEN is not defined, the display does not have a pen.

Domain emptystring

Access Reference

Default Not applicable

4.1.9 PLATFORM dattribute

Definition PLATFORM identifies the name and version identifier of the underlying windowing platform.

The versionid indicates the version of the underlying windowing platform.

If there is no underlying windowing platform, platformversion is equal to implementationvalue.

Domain	<u>platformversion</u>
Access	Reference
Default	Not applicable

4.1.10 PTR dattribute

Definition	If PTR is defined, the display has a pointer, such as a mouse. If PTR is not defined, the display does not have a pointer.
Domain	<u>emptystring</u>
Access	Reference
Default	Not applicable

4.1.11 SIZE dattribute

Definition	SIZE specifies the size of the display expressed in the unit of measure defined by the display's UNITS <u>dattribute</u> .
Domain	<u>size</u>
Access	Reference
Default	Not applicable

4.1.12 SPECTRUM dattribute

Definition	If the value of the display's COLORTYPE <u>dattribute</u> is equal to "COLOR", SPECTRUM specifies the number of distinct color values (that is, the number of combinations of <u>rcolor</u> , <u>gcolor</u> and <u>bcolor</u> values that produce distinct colors) supported by the display. If the value of the display's COLORTYPE <u>dattribute</u> is equal to "GREY", SPECTRUM specifies the number of distinct grey-scale values supported by the display. If the value of the display's COLORTYPE <u>dattribute</u> is equal to "MONO", SPECTRUM is equal to 2.
Domain	<u>intexpr</u>
Access	Reference
Default	Not applicable

4.1.13 TYPEFACE dattribute

Definition	TYPEFACE provides, via descendant nodes, information about the type faces available for the display. Each type face is identified by an entry of the form
------------	---

$$^{\wedge}\$DI[SPLAY](\underline{display}, \underline{tfacekeyword}, \underline{fontface}, \underline{fontsize})$$

where

$$\underline{tfacekeyword} ::= \underline{expr} \vee \text{TYPEFACE}$$

and where fontface and fontsize subscripts indicate available type face and size combinations. A fontsize value of 0 (zero) indicates a scalable type face.

No value is present at the level of the TYPEFACE dattribute itself. For all descendant nodes, the value is an emptystring.

Domain	<u>emptystring</u>
Access	Reference
Default	Not applicable

4.1.14 UNITS dattribute

Definition UNITS indicates the measurement units for the display. If the unit of measure is CHAR, the basis font is the font specified by an FFACE value of "M,DEFAULT", an FSIZE value of 12, and an FSTYLE value of "NORMAL".

Domain	<u>unitspec</u>
Access	Modify, Reference
Default	"PIXEL"

5 Windows

The ssvn ^\$WINDOW contains information about the windows accessible to a process, in the form

^\$W[INDOW] (wname , wattribute)

wname is the identifier for a window.

5.1 Window Types

The MWAPI supports the following window types.

Application windows are the basic input/output mechanism of the MWAPI. Application windows can contain elements.

MTERM windows are terminal emulation windows that accept M input/output commands. The creation of elements within MTERM windows is reserved.

A window's type is indicated by a windowtype value.

windowtype ::= expr V

APPLICATION
MTERM
<u>implementationvalue</u>