**M Windowing API Version 1.2**

**DRAFT**

**February 7, 1996**

**Contents**

1. Fundamental Concepts [1](#_Toc353850818)

1.1 Scope, Application and Use [1](#_Toc353850819)

1.2 Relation to Environment [1](#_Toc353850820)

1.3 Windows, Elements and Choices [1](#_Toc353850821)

1.4 Displays [2](#_Toc353850822)

1.5 Events [2](#_Toc353850823)

1.6 Attributes [2](#_Toc353850824)

1.7 Drawing [2](#_Toc353850825)

1.8 Parents and Children [2](#_Toc353850826)

1.9 Data Exchange [3](#_Toc353850827)

1.10 Build Mode [3](#_Toc353850828)

2. Normative References [4](#_Toc353850829)

3. Attributes [6](#_Toc353850830)

3. [6](#_Toc353850831)

3.1 Manipulating Attributes [6](#_Toc353850832)

3.2 Attribute Definitions [7](#_Toc353850833)

3.3 Attribute Value Domains [8](#_Toc353850834)

3.3.1 Length of Character Strings [8](#_Toc353850835)

3.3.2 Colors [8](#_Toc353850836)

3.3.3 Characters and the Empty String [9](#_Toc353850837)

3.3.4 True/False Values [9](#_Toc353850838)

3.3.5 Implementation and Application Attributes [10](#_Toc353850839)

3.3.6 External Resources [10](#_Toc353850840)

3.3.7 Fonts [11](#_Toc353850841)

3.3.8 Identifiers [12](#_Toc353850842)

3.3.8 [12](#_Toc353850843)

3.3.9 Position, Size and Units [13](#_Toc353850844)

3.3.10 Display Characteristics [15](#_Toc353850845)

3.3.11 Drawing Characteristics [16](#_Toc353850846)

3.3.12 Orientation [17](#_Toc353850847)

3.3.13 Key, Character and Pointer Button Codes [17](#_Toc353850848)

3.3.14 Insertion Point and Selection Range [20](#_Toc353850849)

3.3.14 [20](#_Toc353850850)

3.3.14 [20](#_Toc353850851)

3.3.15 Title Position [20](#_Toc353850852)

3.3.15 [20](#_Toc353850853)

3.3.16 Choice Layout [20](#_Toc353850854)

3.3.16 [20](#_Toc353850855)

3.3.17 Modality [21](#_Toc353850856)

3.3.17 [21](#_Toc353850857)

3.3.18 Scrolling [21](#_Toc353850858)

3.3.18 [21](#_Toc353850859)

3.3.19 Focus Location [22](#_Toc353850860)

3.3.20 Data Exchange [22](#_Toc353850861)

3.3.21 Build Mode [23](#_Toc353850862)

4. Displays [24](#_Toc353850863)

4.1 Display Attributes [24](#_Toc353850864)

4.1.1 BCOLOR dattribute [25](#_Toc353850865)

4.1.2 CLIPBOARD dattribute [25](#_Toc353850866)

4.1.2 [25](#_Toc353850867)

4.1.3 COLOR dattribute [25](#_Toc353850868)

4.1.3 [25](#_Toc353850869)

4.1.4 COLORTYPE dattribute [25](#_Toc353850870)

4.1.5 FCOLOR dattribute [26](#_Toc353850871)

4.1.6 FFACE dattribute [26](#_Toc353850872)

4.1.7 FOCUS dattribute [26](#_Toc353850873)

4.1.7 [26](#_Toc353850874)

4.1.8 FSIZE dattribute [27](#_Toc353850875)

4.1.9 FSTYLE dattribute [27](#_Toc353850876)

4.1.10 KEYBOARD dattribute [28](#_Toc353850877)

4.1.10 [28](#_Toc353850878)

4.1.11 LINKID dattribute [28](#_Toc353850879)

4.1.11 [28](#_Toc353850880)

4.1.12 LINKSERVICE dattribute [28](#_Toc353850881)

4.1.12 [28](#_Toc353850882)

4.1.13 LINKSOURCE dattribute [29](#_Toc353850883)

4.1.13 [29](#_Toc353850884)

4.1.14 PEN dattribute [29](#_Toc353850885)

4.1.14 [29](#_Toc353850886)

4.1.15 PLATFORM dattribute [29](#_Toc353850887)

4.1.16 PTR dattribute [30](#_Toc353850888)

4.1.17 PTYPE dattribute [30](#_Toc353850889)

4.1.18 SIZE dattribute [31](#_Toc353850890)

4.1.18 [31](#_Toc353850891)

4.1.19 SPECTRUM dattribute [31](#_Toc353850892)

4.1.20 TBCOLOR dattribute [31](#_Toc353850893)

4.1.21 TFCOLOR dattribute [32](#_Toc353850894)

4.1.22 TYPEFACE dattribute [32](#_Toc353850895)

4.1.23 UNITS dattribute [32](#_Toc353850896)

4.1.23 [32](#_Toc353850897)

5. Windows [34](#_Toc353850898)

5.1 Window Types [34](#_Toc353850899)

5.1.1 MTERM Windows [34](#_Toc353850900)

5.2 Window Components [35](#_Toc353850901)

5.3 Window Repainting [35](#_Toc353850902)

5.4 Window Attributes [35](#_Toc353850903)

5.4.1 Applicability of Window Attributes [36](#_Toc353850904)

5.4.2 ACTIVE wattribute [38](#_Toc353850905)

5.4.2 [38](#_Toc353850906)

5.4.3 BCOLOR wattribute [38](#_Toc353850907)

5.4.4 BEVENT wattribute [38](#_Toc353850908)

5.4.4 [38](#_Toc353850909)

5.4.5 BLDMODE wattribute [39](#_Toc353850910)

5.4.5 [39](#_Toc353850911)

5.4.6 COLOR wattribute [39](#_Toc353850912)

5.4.6 [39](#_Toc353850913)

5.4.7 DEFBUTTON wattribute [40](#_Toc353850914)

5.4.8 DISPLAY wattribute [40](#_Toc353850915)

5.4.8 [40](#_Toc353850916)

5.4.9 EVENT wattribute [41](#_Toc353850917)

5.4.9 [41](#_Toc353850918)

5.4.10 FCOLOR wattribute [41](#_Toc353850919)

5.4.11 FFACE wattribute [41](#_Toc353850920)

5.4.12 FSIZE wattribute [42](#_Toc353850921)

5.4.13 FSTYLE wattribute [42](#_Toc353850922)

5.4.14 GRID wattribute [43](#_Toc353850923)

5.4.14 [43](#_Toc353850924)

5.4.15 ICON wattribute [44](#_Toc353850925)

5.4.15 [44](#_Toc353850926)

5.4.16 ICONIFY wattribute [44](#_Toc353850927)

5.4.16 [44](#_Toc353850928)

5.4.17 ID wattribute [44](#_Toc353850929)

5.4.18 ITITLE wattribute [45](#_Toc353850930)

5.4.19 LINKTOPIC wattribute [45](#_Toc353850931)

5.4.20 MAX wattribute [45](#_Toc353850932)

5.4.21 MENUBAR wattribute [46](#_Toc353850933)

5.4.22 MIN wattribute [46](#_Toc353850934)

5.4.23 MODAL wattribute [46](#_Toc353850935)

5.4.23 [46](#_Toc353850936)

5.4.24 NEXTG wattribute [47](#_Toc353850937)

5.4.25 PARENT wattribute [47](#_Toc353850938)

5.4.25 [47](#_Toc353850939)

5.4.26 POS wattribute [48](#_Toc353850940)

5.4.26 [48](#_Toc353850941)

5.4.27 PSELECT wattribute [49](#_Toc353850942)

5.4.27 [49](#_Toc353850943)

5.4.28 PTYPE wattribute [49](#_Toc353850944)

5.4.28 [49](#_Toc353850945)

5.4.29 RESIZE wattribute [50](#_Toc353850946)

5.4.29 [50](#_Toc353850947)

5.4.30 SCROLL wattribute [50](#_Toc353850948)

5.4.30 [50](#_Toc353850949)

5.4.31 SIZE wattribute [50](#_Toc353850950)

5.4.31 [50](#_Toc353850951)

5.4.32 SIZEMIN wattribute [51](#_Toc353850952)

5.4.33 SIZEWIN wattribute [52](#_Toc353850953)

5.4.34 TBCOLOR wattribute [52](#_Toc353850954)

5.4.35 TFCOLOR wattribute [53](#_Toc353850955)

5.4.36 TIED wattribute [53](#_Toc353850956)

5.4.36 [53](#_Toc353850957)

5.4.37 TITLE wattribute [53](#_Toc353850958)

5.4.37 [53](#_Toc353850959)

5.4.38 TYPE wattribute [54](#_Toc353850960)

5.4.38 [54](#_Toc353850961)

5.4.39 UNITS wattribute [54](#_Toc353850962)

5.4.39 [54](#_Toc353850963)

5.4.40 VALUE wattribute [54](#_Toc353850964)

5.4.40 [54](#_Toc353850965)

5.4.41 VISIBLE wattribute [55](#_Toc353850966)

5.4.41 [55](#_Toc353850967)

6. Elements [56](#_Toc353850968)

6.1 Gadgets [56](#_Toc353850969)

6.1.1 Check Box [56](#_Toc353850970)

6.1.2 Document [57](#_Toc353850971)

6.1.3 Generic Box [57](#_Toc353850972)

6.1.4 Group Frame [57](#_Toc353850973)

6.1.5 Label [57](#_Toc353850974)

6.1.5 [57](#_Toc353850975)

6.1.6 List Box [57](#_Toc353850976)

6.1.7 List Button [58](#_Toc353850977)

6.1.8 List Entry Box [58](#_Toc353850978)

6.1.9 Long List Box [58](#_Toc353850979)

6.1.10 Push Button [58](#_Toc353850980)

6.1.11 Radio Button Set [58](#_Toc353850981)

6.1.12 Scroll [59](#_Toc353850982)

6.1.12 [59](#_Toc353850983)

6.1.13 Symbol [59](#_Toc353850984)

6.1.14 Table [59](#_Toc353850985)

6.1.15 Text [59](#_Toc353850986)

6.2 Menus [60](#_Toc353850987)

6.3 Timers [60](#_Toc353850988)

6.4 Element Attributes [61](#_Toc353850989)

6.4.1 Applicability of Element Attributes [62](#_Toc353850990)

6.4.2 ACTIVE elattribute [69](#_Toc353850991)

6.4.2 [69](#_Toc353850992)

6.4.3 BCOLOR elattribute [69](#_Toc353850993)

6.4.4 BLDMODE elattribute [69](#_Toc353850994)

6.4.4 [69](#_Toc353850995)

6.4.5 CANCEL elattribute [70](#_Toc353850996)

6.4.5 [70](#_Toc353850997)

6.4.6 CANCHANGE elattribute [70](#_Toc353850998)

6.4.7 CHANGED elattribute [70](#_Toc353850999)

6.4.7 [70](#_Toc353851000)

6.4.8 CHARMAX elattribute [71](#_Toc353851001)

6.4.9 CHOICE elattribute [71](#_Toc353851002)

6.4.9 [71](#_Toc353851003)

6.4.10 COL elattribute [72](#_Toc353851004)

6.4.10 [72](#_Toc353851005)

6.4.11 COLWIDTH elattribute [73](#_Toc353851006)

6.4.11 [73](#_Toc353851007)

6.4.12 DRAW elattribute [73](#_Toc353851008)

6.4.13 DRAWTYPE elattribute [74](#_Toc353851009)

6.4.14 EVENT elattribute [74](#_Toc353851010)

6.4.14 [74](#_Toc353851011)

6.4.15 FCOLOR elattribute [74](#_Toc353851012)

6.4.16 FFACE elattribute [75](#_Toc353851013)

6.4.17 FRAMED elattribute [75](#_Toc353851014)

6.4.18 FSIZE elattribute [75](#_Toc353851015)

6.4.19 FSTYLE elattribute [75](#_Toc353851016)

6.4.20 GRID elattribute [76](#_Toc353851017)

6.4.21 HCOLID elattribute [76](#_Toc353851018)

6.4.22 HROWID elattribute [76](#_Toc353851019)

6.4.23 ID elattribute [77](#_Toc353851020)

6.4.24 INSELECT elattribute [77](#_Toc353851021)

6.4.25 INTERVAL elattribute [78](#_Toc353851022)

6.4.25 [78](#_Toc353851023)

6.4.26 LEFTSHOW elattribute [78](#_Toc353851024)

6.4.27 LINKACTION elattribute [79](#_Toc353851025)

6.4.28 LINKITEM elattribute [80](#_Toc353851026)

6.4.29 LINKMODE elattribute [80](#_Toc353851027)

6.4.30 LINKSOURCE elattribute [80](#_Toc353851028)

6.4.31 LINKTIMEOUT elattribute [81](#_Toc353851029)

6.4.32 NEXTG elattribute [81](#_Toc353851030)

6.4.33 POS elattribute [82](#_Toc353851031)

6.4.33 [82](#_Toc353851032)

6.4.34 PTYPE elattribute [82](#_Toc353851033)

6.4.34 [82](#_Toc353851034)

6.4.35 RESOURCE elattribute [83](#_Toc353851035)

6.4.35 [83](#_Toc353851036)

6.4.36 ROW elattribute [83](#_Toc353851037)

6.4.36 [83](#_Toc353851038)

6.4.37 ROWCOL elattribute [84](#_Toc353851039)

6.4.37 [84](#_Toc353851040)

6.4.38 ROWHEIGHT elattribute [85](#_Toc353851041)

6.4.38 [85](#_Toc353851042)

6.4.39 SCROLL elattribute [85](#_Toc353851043)

6.4.39 [85](#_Toc353851044)

6.4.40 SCROLLBY elattribute [86](#_Toc353851045)

6.4.40 [86](#_Toc353851046)

6.4.41 SCROLLDIR elattribute [86](#_Toc353851047)

6.4.42 SCROLLPOS elattribute [87](#_Toc353851048)

6.4.43 SCROLLRANGE elattribute [87](#_Toc353851049)

6.4.43 [87](#_Toc353851050)

6.4.44 SELECTMAX elattribute [88](#_Toc353851051)

6.4.45 SELECTVAL elattribute [88](#_Toc353851052)

6.4.46 SIZE elattribute [88](#_Toc353851053)

6.4.46 [88](#_Toc353851054)

6.4.47 TBCOLOR elattribute [90](#_Toc353851055)

6.4.48 TFCOLOR elattribute [90](#_Toc353851056)

6.4.49 TFFACE elattribute [91](#_Toc353851057)

6.4.50 TFSIZE elattribute [91](#_Toc353851058)

6.4.51 TFSTYLE elattribute [91](#_Toc353851059)

6.4.52 TITLE elattribute [91](#_Toc353851060)

6.4.52 [91](#_Toc353851061)

6.4.53 TOPSHOW elattribute [92](#_Toc353851062)

6.4.54 TPOS elattribute [93](#_Toc353851063)

6.4.55 TYPE elattribute [94](#_Toc353851064)

6.4.55 [94](#_Toc353851065)

6.4.56 UNITS elattribute [94](#_Toc353851066)

6.4.56 [94](#_Toc353851067)

6.4.57 VALUE elattribute [94](#_Toc353851068)

6.4.57 [94](#_Toc353851069)

6.4.58 VISIBLE elattribute [96](#_Toc353851070)

6.5 Choice Attributes [97](#_Toc353851071)

6.5.1 Applicability of Choice Attributes [98](#_Toc353851072)

6.5.2 ACCELERATOR cattribute [98](#_Toc353851073)

6.5.2 [98](#_Toc353851074)

6.5.3 ACTIVE cattribute [98](#_Toc353851075)

6.5.3 [98](#_Toc353851076)

6.5.4 AID cattribute [99](#_Toc353851077)

6.5.5 EVENT cattribute [99](#_Toc353851078)

6.5.5 [99](#_Toc353851079)

6.5.6 MARKER cattribute [100](#_Toc353851080)

6.5.7 RESOURCE cattribute [100](#_Toc353851081)

6.5.8 SEPARATOR cattribute [100](#_Toc353851082)

6.5.9 SUBMENU cattribute [101](#_Toc353851083)

6.5.9 [101](#_Toc353851084)

6.6 Row and Column Attributes [101](#_Toc353851085)

6.6.1 Applicability of Row and Column Attributes [101](#_Toc353851086)

6.6.2 COLWIDTH colattribute [102](#_Toc353851087)

6.6.2 [102](#_Toc353851088)

6.6.3 ROWHEIGHT rowattribute [102](#_Toc353851089)

6.6.3 [102](#_Toc353851090)

7. Drawing [103](#_Toc353851091)

7.1 Draw Command Numbers [103](#_Toc353851092)

7.2 Draw Commands [104](#_Toc353851093)

7.2.1 ARC Draw Command [105](#_Toc353851094)

7.2.1 [105](#_Toc353851095)

7.2.2 BCOLOR Draw Command [105](#_Toc353851096)

7.2.3 BITMAP Draw Command [106](#_Toc353851097)

7.2.4 BOX Draw Command [106](#_Toc353851098)

7.2.4 [106](#_Toc353851099)

7.2.5 DMODE Draw Command [106](#_Toc353851100)

7.2.6 DRAWTEXT Draw Command [107](#_Toc353851101)

7.2.7 FCOLOR Draw Command [107](#_Toc353851102)

7.2.8 FFACE Draw Command [107](#_Toc353851103)

7.2.9 FILLPAT Draw Command [107](#_Toc353851104)

7.2.10 FSIZE Draw Command [108](#_Toc353851105)

7.2.11 FSTYLE Draw Command [109](#_Toc353851106)

7.2.12 LINE Draw Command [109](#_Toc353851107)

7.2.12 [109](#_Toc353851108)

7.2.13 OVAL Draw Command [109](#_Toc353851109)

7.2.14 PENSIZE Draw Command [109](#_Toc353851110)

7.2.15 PICTURE Draw Command [110](#_Toc353851111)

7.2.16 PIE Draw Command [110](#_Toc353851112)

7.2.17 POINT Draw Command [110](#_Toc353851113)

7.2.17 [110](#_Toc353851114)

7.2.18 UNITS Draw Command [110](#_Toc353851115)

7.2.18 [110](#_Toc353851116)

8. Data Exchange [111](#_Toc353851117)

8.1 Server [111](#_Toc353851118)

8.2 Client [111](#_Toc353851119)

9. Events [112](#_Toc353851120)

9.1 Event Types [112](#_Toc353851121)

9.1.1 Window State Events [113](#_Toc353851122)

9.1.2 Pointer Events [114](#_Toc353851123)

9.1.3 Keyboard Events [115](#_Toc353851124)

9.1.4 Focus Events [115](#_Toc353851125)

9.1.5 Select Events [116](#_Toc353851126)

9.1.6 Long List Box Events [117](#_Toc353851127)

9.1.7 Help Events [117](#_Toc353851128)

9.1.8 Timer Events [118](#_Toc353851129)

9.1.9 Data Exchange Events [118](#_Toc353851130)

9.1.10 Build Mode Events [119](#_Toc353851131)

9.1.11 Applicability of Event Types [119](#_Toc353851132)

9.2 Event Specification Attributes [127](#_Toc353851133)

9.2.1 ENABLE especattribute [128](#_Toc353851134)

9.2.1 [128](#_Toc353851135)

9.2.2 FILTERIN especattribute [128](#_Toc353851136)

9.3 Processing Events [129](#_Toc353851137)

9.4 Event Commands [129](#_Toc353851138)

9.4.1 ESTART [129](#_Toc353851139)

9.4.2 ESTOP [130](#_Toc353851140)

9.4.3 ETRIGGER [130](#_Toc353851141)

9.5 Event Information [131](#_Toc353851142)

9.5.1 CHOICE einfoattribute [132](#_Toc353851143)

9.5.1 [132](#_Toc353851144)

9.5.2 CLASS einfoattribute [132](#_Toc353851145)

9.5.3 COL einfoattribute [133](#_Toc353851146)

9.5.4 ELEMENT einfoattribute [133](#_Toc353851147)

9.5.4 [133](#_Toc353851148)

9.5.5 GROUP einfoattribute [133](#_Toc353851149)

9.5.6 KEY einfoattribute [134](#_Toc353851150)

9.5.6 [134](#_Toc353851151)

9.5.7 LINKCOMMAND einfoattribute [134](#_Toc353851152)

9.5.8 LINKID einfoattribute [134](#_Toc353851153)

9.5.9 LINKITEM einfoattribute [135](#_Toc353851154)

9.5.10 LINKREASON einfoattribute [135](#_Toc353851155)

9.5.11 LINKSOURCE einfoattribute [135](#_Toc353851156)

9.5.12 LINKSTATUS einfoattribute [135](#_Toc353851157)

9.5.13 NEXTFOCUS einfoattribute [136](#_Toc353851158)

9.5.14 OK einfoattribute [136](#_Toc353851159)

9.5.15 PBUTTON einfoattribute [136](#_Toc353851160)

9.5.16 PPOS einfoattribute [137](#_Toc353851161)

9.5.17 PRIORFOCUS einfoattribute [137](#_Toc353851162)

9.5.18 PRIORPOS einfoattribute [137](#_Toc353851163)

9.5.19 PSELECT einfoattribute [138](#_Toc353851164)

9.5.20 PSTATE einfoattribute [138](#_Toc353851165)

9.5.21 ROW einfoattribute [138](#_Toc353851166)

9.5.22 SEQUENCE einfoattribute [139](#_Toc353851167)

9.5.22 [139](#_Toc353851168)

9.5.23 TRACKPOS einfoattribute [139](#_Toc353851169)

9.5.24 TRACKSIZE einfoattribute [139](#_Toc353851170)

9.5.25 TYPE einfoattribute [139](#_Toc353851171)

9.5.25 [139](#_Toc353851172)

9.5.26 WINDOW einfoattribute [140](#_Toc353851173)

9.5.26 [140](#_Toc353851174)

9.5.27 Applicability of Event Information Attributes [140](#_Toc353851175)

10. Additional Functions and Special Variables [155](#_Toc353851176)

10.1 Functions [155](#_Toc353851177)

10.1.1 $WFONT [155](#_Toc353851178)

10.1.2 $WTFIT [156](#_Toc353851179)

10.1.3 $WTWIDTH [156](#_Toc353851180)

10.2 Special Variables [156](#_Toc353851181)

11. Error Handling [157](#_Toc353851182)

11.1 Error Information [157](#_Toc353851183)

11.2 Errors [158](#_Toc353851184)

12. Conformance [159](#_Toc353851185)

12.1 Implementations [159](#_Toc353851186)

12.1.1 Conforming Implementation [159](#_Toc353851187)

12.1.2 MDC Conforming Implementation [162](#_Toc353851188)

12.1.3 MDC Strictly Conforming Implementation [162](#_Toc353851189)

12.1.4 MDC Partial Implementation [163](#_Toc353851190)

12.1.5 Multiple Levels of Conformance [163](#_Toc353851191)

**Tables**

**Window Attributes** [37](#_Toc353851193)

**Element Attributes** [63](#_Toc353851194)

**Choice Attributes** [98](#_Toc353851195)

**Events** [120](#_Toc353851196)

**Event Information** [141](#_Toc353851197)

**Event Information** [142](#_Toc353851198)

**Implementation Definition** [159](#_Toc353851199)

**Implementation-Specific Features** [160](#_Toc353851200)

**Portability Limits** [161](#_Toc353851201)

**Unimplemented Features** [162](#_Toc353851202)

# Fundamental Concepts

* 1. Scope, Application and Use

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 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. 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. 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. 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. 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. 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. Drawing

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

* 1. 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 (B1 ... Bn) such that A is the parent of B1, Bn is the parent of C, and, for x equal to 1 through n-1, Bx is the parent of Bx+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.

* 1. Data Exchange

Using the Data Exchange capabilities of MWAPI an M process can access items in a data source and can itself provide data sources. See Section 8 for more details.

* 1. Build Mode

To support window layout tools windows may be created in build mode. The behavior of a window and its elements is modified in this mode. For example gadgets may be moved and resized but do not respond to normal user input, and timers do not run. A different set of events apply in build mode, all originated by the window. Selection, movement and resizing of one or more gadgets is supported in a platform-specific fashion by means of the pointer and/or keyboard.

# 

# Normative References

The following standard contains provisions which, through reference in this text, constitute provisions of this American National Standard. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this American National Standard are encouraged to investigate the possibility of applying the most recent edition of the standard listed below.

ANSI/MDC X11.1-1995, American national standard for information systems -- programming languages -- MUMPS.

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

actuallist

command

control

controlmnemonic

digit

doargument

emptystring

eol

expr

expratom

function

glvn

graphic

gvn

ident

intexpr

intlit

killargument

L

lvn

mnemonicspace

name

namevalue

numexpr

numlit

openargument

postcond

process

routine

setleft

SP

ssvn

structured system variable

subscript

svn

timeout

tvexpr

V

xargument

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

# 

# 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, rowattribute, colattribute, especattribute, and einfoattribute values, and are referred to collectively as attribute names. All attribute names not specified by this 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.

* 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 setleft 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 an ecode = "M46" 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 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 an ecode = "M47" 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.

* 1. 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, rowattribute, colattribute, 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. The attribute node may be undefined.

Not

Applicable Indicates that no default applies because the attribute can only be referenced.

* 1. Attribute Value Domains
     1. Length of Character Strings

When the MWAPI is present, the portability requirement in Section 2 of the MUMPS 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 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 Standard.

* + 1. Colors

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

|  |  |
| --- | --- |
| color ::= expr V | rcolor , gcolor , bcolor  emptystring |

rcolor ::= intlit

gcolor ::= intlit

bcolor ::= intlit

A color value of the form

rcolor , gcolor , bcolor

identifies a specific color. 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.

A color value of the form emptystring indicates that the appropriate default color from the underlying windowing platform should be used. If the underlying windowing platform does not provide an appropriate default, the color is determined by the MWAPI implementation.

* + 1. Characters and the Empty String

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| chars | [ | control  graphic | ] | ... |

longchars ::= chars

tab ::= a tab character

linend ::= eol

mnemonichar ::= an ampersand character

* + 1. 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.

* + 1. 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.

### 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.

|  |  |  |  |
| --- | --- | --- | --- |
|  | extresource | | |
| symbol ::= | expr V | 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.

|  |  |  |  |
| --- | --- | --- | --- |
|  | extresource | | |
| marker ::= | expr V | 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.

|  |  |  |  |
| --- | --- | --- | --- |
|  | extresource | | |
| ptype ::= | expr V | M,CROSS  M,IBEAM  M,WAIT M,POINTER |  |

ptype values of "M,CROSS", "M,IBEAM", "M,WAIT", and “M,POINTER” specify pointer appearance that indicates cross-hairs, an I-beam, waiting, and standard pointer, 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.

* + 1. Fonts

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

|  |  |
| --- | --- |
| fontface ::= expr V | M,DEFAULT  M,FIXED  M,SANS  M,SERIF  alpha chars |

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.

|  |  |
| --- | --- |
| fontstyle ::= expr V | NORMAL  fstyle [,fstyle] ... |

|  |  |
| --- | --- |
| fstyle ::= | BOLD  ITALIC  ULINE  implementationvalue |

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. Except for in drawcommands, any references to attributes identifying fonts (including attributes to which values beginning with "M," have been assigned) yield information about the actual font.

* + 1. Identifiers

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

|  |  |  |
| --- | --- | --- |
| wident ::= | %  alphanum | [ alphanum ] ... |

|  |  |
| --- | --- |
| alphanum ::= | ident  digit |

Windows are identified by wnames.

wname ::= wident

Elements are identified by enames.

|  |  |
| --- | --- |
| ename ::= | gname  mname  tname |

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

ritem ::= item

citem ::= item

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 an ecode = "M48" 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.

* + 1. 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.

|  |  |
| --- | --- |
| unitspec ::= | CHAR [ , chars ]  PIXEL  POINT  REL [ , hscale , vscale ]  implementationvalue |

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:

hpix = hrel / hscale \* hbasis

vpix = vrel / vscale \* 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 ] ]  , 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).

|  |  |
| --- | --- |
| size ::= expr V | hsize [ , [ vsize ] [ , unitspec ] ]  , vsize [ , unitspec ] |

hsize ::= numlit

vsize ::= numlit

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

width ::= hsize [ , unitspec ]

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

height ::= vsize [ , unitspec ]

Unless indicated otherwise, if an application attempts to assign to an attribute an hsize or vsize value which is less than zero, an error occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If the unitspec is omitted when a pos, size, width, or height 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, size, width, or height value yields a value that includes a unitspec piece and that is expressed in the unit of measure specified (when the reference is made) by the entity's UNITS attribute. Modification of a UNITS attribute does not cause the actual size or position of any entity to change.

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

* + 1. Display Characteristics

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

|  |  |
| --- | --- |
| displaycolor ::= expr V | COLOR  GREY  MONO  implementationvalue |

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 ]

|  |  |
| --- | --- |
| platformname ::= | MAC  MSWIN  PM  X/MTF  X/OPNLK  implementationvalue |

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

* + 1. Drawing Characteristics

drawtype specifies the type of drawcommands that can be specified.

|  |  |
| --- | --- |
| drawtype ::= expr V | MDRAW  implementationvalue |

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

|  |  |
| --- | --- |
| fillmode ::= expr V | OPAQUE  TRANSPARENT  implementationvalue |

fillpattern specifies the pattern used to fill an area.

|  |  |
| --- | --- |
| fillpattern ::= expr V | BDIAGONAL  CROSSHATCH  DCROSSHATCH  FDIAGONAL  HORIZONTAL  NONE  SOLID  VERTICAL  implementationvalue |

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

|  |  |
| --- | --- |
| dmode ::= expr V | SOLID  XOR  implementationvalue |

* + 1. 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.

|  |  |
| --- | --- |
| orientation ::= expr V | H  V  implementationvalue |

* + 1. Key, Character and Pointer Button Codes

Keyboard keys are identified by *key codes* (keycodes).

|  |  |
| --- | --- |
| keycode ::= | graphic  commacode  functionkey  modifierkey  implementationvalue |

commacode ::= COMMA

|  |  |
| --- | --- |
| modifierkey ::= | ALT  ALTR  COMMAND  COMPOSE  COMPOSER  CTRL  CTRLR  OPTION  SHIFT  SHIFTR |
| functionkey ::= | BACKSPC  BREAK  CAPSLOCK  DEL  DOWN  END  ENTER  ESC  F1  F2  F3  F4  F5  F6  F7  F8  F9  F10  F11  F12  F13  F14  F15  F16  F17  F18  F19  F20  F21  F22  F23  F24  FIND  HOME  INS  LEFT  NEXT |

|  |  |
| --- | --- |
|  | NUM0  NUM1  NUM2  NUM3  NUM4  NUM5  NUM6  NUM7  NUM8  NUM9  NUMDEC  NUMDEL  NUMDIV  NUMENTER  NUMLOCK  NUMMINUS  NUMMULT  NUMPLUS  PAGEDOWN  PAGEUP  PAUSE  PF1  PF2  PF3  PF4  RET  RIGHT  SCRLLOCK  SEL  TAB  UP |

Key sequences, consisting of one or more keys, which result in the production of a character are identified by *character codes* (charcodes).

|  |  |
| --- | --- |
| charcode ::= [ modifierkey + ] ... | graphic  commacode  functionkey  implementationvalue |

A charcode never contains SHIFT (or SHIFTR) in combination with a graphic.

The comma character has two representations: the comma graphic and commacode. When an application assigns a value to an attribute that has as its domain a list of key codes (L keycode) or a list of character codes (L charcode), the commacode form must be used. When the MWAPI implementation sets the value of an attribute with a domain of keycode or charcode, the graphic form is used.

Pointer buttons are identified by *pointer button code*s (pbuttoncodes).

|  |  |
| --- | --- |
| pbuttoncode ::= [ modifierkey + ] ... | PB1  PB2  PB3  implementationvalue |

If a pointing device has a single button, it is identified as PB1. If a pointing device has two buttons, they are identified as PB1 and PB3. If a pointing device has three buttons, they are identified as PB1, PB2 and PB3. For any pointing device, PB1 indicates the primary button.

When the MWAPI implementation assigns to an attribute a charcode or pbuttoncode that contains multiple modifierkeys, the modifierkeys are listed in collating sequence. When an application assigns to an attribute a charcode or pbuttoncode that contains multiple modifierkeys, the modifierkeys can be listed in any sequence.

* + 1. Insertion Point and Selection Range

insel ::= expr V insert , seloff

insert ::= intlit

seloff ::= [ - ] intlit

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.

* + 1. Title Position

titlepos specifies where a gadget title appears.

|  |  |
| --- | --- |
| titlepos ::= expr V | LEFT  RIGHT  TOP  implementationattribute |

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.

* + 1. Choice Layout

rowcolspec specifies how choices are positioned within a gadget.

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

|  |  |
| --- | --- |
| rowcol ::= | COL  ROW |

rowcolsize ::= intlit

|  |  |
| --- | --- |
| filldir ::= | H  V |

The meaning of rowcolspec is specified in Section 6.4.37

* + 1. Modality

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

|  |  |
| --- | --- |
| modaltype ::= expr V | ANCESTORS  APPLICATION  PARENT  implementationvalue |

The meaning of modaltype is specified in Section 5.4.23.

* + 1. Scrolling

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

|  |  |
| --- | --- |
| scrollrange ::= expr V | scrollmin  [ scrollmin ] , scrollmax |

scrollmin ::= numlit

scrollmax ::= numlit

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 | scrollsmall  [ scrollsmall ] , scrollbig |

scrollsmall ::= numlit

scrollbig ::= numlit

* + 1. Focus Location

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

|  |  |
| --- | --- |
| focusloc ::= expr V | wname  wname , gname  emptystring |

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

In general, a window is considered to have focus when it or any of its child gadgets has focus.

* + 1. Data Exchange

deaction identifies an action to perform over a Data Exchange link

|  |  |
| --- | --- |
| deaction ::= expr V | SEND  REQUEST  EXECUTE |

destatus identifies the outcome of an action performed over a Data Exchange link

|  |  |
| --- | --- |
| destatus ::= expr V | FAILED  SUCCESS  TIMEOUT |

delinkid is a platform-specific identifier for a data exchange link. A delinkid must be a valid M subscript, and each link in which an M process is concurrently involved must yield a unique identifier.

delink ::= chars

deitem identifies an item within a data source

deitem ::= chars

demode defines the operating mode of a Data Exchange link

|  |  |
| --- | --- |
| demode ::= expr V | AUTOMATIC  MANUAL  NONE NOTIFY |

derole defines the role of the M process in a Data Exchange link

|  |  |
| --- | --- |
| derole ::= expr V | CLIENT SERVER |

deservice identifies the service name under which a display acts as a data exchange source.

deservice ::= chars

desource identifies a data exchange source available to the M process. Each such desource must be a valid M subscript.

desource ::= chars

detopic identifies the topic name under which a window acts as a data exchange source.

detopic ::= chars

### Build Mode

buildops specifies the behavior of a gadget on a build-mode window.

|  |  |
| --- | --- |
| buildops ::= expr V | emptystring  MOVE  SIZE MOVE,SIZE |

gridattr specifies an attribute of the grid on a build-mode window.

|  |  |
| --- | --- |
| gridattr ::= expr V | SNAPPOS  SNAPSIZE VISIBLE |

# 

# 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.

* 1. Display Attributes

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

|  |  |
| --- | --- |
| dattribute ::= expr V | BCOLOR  CLIPBOARD  COLOR  COLORTYPE  FCOLOR  FFACE  FOCUS  FSIZE  FSTYLE  KEYBOARD LINKID LINKSERVICE LINKSOURCE  PEN  PLATFORM  PTYPE  PTR  SIZE  SPECTRUM  TBCOLOR  TFCOLOR  TYPEFACE  UNITS  applicationattribute  implementationattribute |

* + 1. BCOLOR dattribute

**Definition** BCOLOR indicates the default value for the BCOLOR wattribute for windows that are children of the display.

**Domain** color

**Access** Modify, Reference

**Default** Platform

* + 1. 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** longchars

**Access** Modify, Reference

**Default** Not Applicable

* + 1. COLOR dattribute

**Definition** COLOR indicates the default value for the COLOR wattribute for windows that are children of the display.

**Domain** color

**Access** Modify, Reference

**Default** Platform

* + 1. COLORTYPE dattribute

**Definition** COLORTYPE indicates the color capabilities of the display.

**Domain** displaycolor

**Access** Reference

**Default** Not Applicable

* + 1. FCOLOR dattribute

**Definition** FCOLOR indicates the default value for the FCOLOR wattribute for windows that are children of the display.

**Domain** color

**Access** Modify, Reference

**Default** Platform

* + 1. FFACE dattribute

**Definition** FFACE indicates the default value for the FFACE wattribute for windows that are children of the display.

**Domain** fontface

**Access** Modify, Reference

**Default** "M,DEFAULT"

* + 1. 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 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.

If an application sets focus to a window that is iconified, the window is restored.

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 is incapable of getting focus in the underlying windowing platform, or to set focus to a window that is disabled because of the value of the MODAL wattribute of another window, or to set focus to a window with a DISPLAY wattribute value that identifies a different display, an error condition occurs with an ecode = "M49" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** focusloc

**Access** Modify, Reference

**Default** Platform

* + 1. FSIZE dattribute

**Definition** FSIZE indicates the default value for the FSIZE wattribute for windows that are children of the display.

**Domain** fontsize

**Access** Modify, Reference

**Default** Platform. The font size provided by the platform when requested for a FFACE of “M,DEFAULT”, a FSIZE of 12 and a FSTYLE of “NORMAL”.

* + 1. FSTYLE dattribute

**Definition** FSTYLE indicates the default value for the FSTYLE wattribute for windows that are children of the display.

**Domain** fontstyle

**Access** Modify, Reference

**Default** Platform. The font style provided by the platform when requested for a FFACE of “M,DEFAULT”, a FSIZE of 12 and a FSTYLE of “NORMAL”.

* + 1. 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

* + 1. LINKID dattribute

**Definition** LINKID provides, via descendant nodes, information about the data exchange links in which the display is participating. Each link is identified by an entry of the form

^$DI[SPLAY](display , linkidkeyword , delinkid )

where

linkidkeyword ::= expr V LINKID

No value is present at the level of the LINKID dattribute itself. For all descendant nodes, the value indicates the role of the M process in the link. This value may not be set, but may be killed. By killing a delinkid subscript an M process closes the corresponding link.

**Domain** derole

**Access** Modify, Reference

**Default** Not applicable

* + 1. LINKSERVICE dattribute

**Definition** LINKSERVICE specifies the service name under which the M process offers any sources defined by LINKTOPIC wattributes on windows on the display. When LINKSERVICE is undefined, none of the sources are offered. When LINKSERVICE is defined it combines with LINKTOPIC wattributes in a platform-specific manner to give the desources of the data exchange sources offered by the M process.

**Domain** deservice

**Access** Create, Modify, Reference

**Default** None

* + 1. LINKSOURCE dattribute

**Definition** LINKSOURCE provides, via descendant nodes, information about the data exchange sources available for the display. Each source is identified by an entry of the form

^$DI[SPLAY](display , linksourcekeyword , desource )

where

linksourcekeyword ::= expr V LINKSOURCE

and where desource subscripts indicate available data exchange sources.

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

**Domain** emptystring

**Access** Reference

**Default** Not applicable

* + 1. 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

* + 1. 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** platformversion

**Access** Reference

**Default** Not applicable

* + 1. 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** emptystring

**Access** Reference

**Default** Not applicable

* + 1. PTYPE dattribute

**Definition** If PTYPE is defined, it specifies the pointer appearance when the pointer is over the viewport of any window defined in ^$WINDOW which appears on the display. The PTYPE dattribute takes precedence over all PTYPE wattributes and over all PTYPE elattributes.

If the PTYPE dattribute is not defined and the pointer is not over a window or gadget for which a PTYPE wattribute or elattribute, respectively, is defined, the pointer appearance is determined by the platform.

**Domain** ptype

**Access** Modify, Reference

**Default** Not applicable

* + 1. SIZE dattribute

**Definition** SIZE specifies the size of the display expressed in the unit of measure defined by the display's UNITS dattribute.

**Domain** size

**Access** Reference

**Default** Not applicable

* + 1. SPECTRUM dattribute

**Definition** If the value of the display's COLORTYPE dattribute is equal to "COLOR", SPECTRUM specifies the number of distinct color values (that is, the number of combinations of rcolor, gcolor and bcolor values that produce distinct colors) supported by the display. If the value of the display's COLORTYPE dattribute 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 dattribute is equal to "MONO", SPECTRUM is equal to 2.

**Domain** intexpr

**Access** Reference

**Default** Not applicable

* + 1. TBCOLOR dattribute

**Definition** TBCOLOR indicates the default value for the TBCOLOR wattribute for windows that are children of the display.

**Domain** color

**Access** Modify, Reference

**Default** None

* + 1. TFCOLOR dattribute

**Definition** TFCOLOR indicates the default value for the TFCOLOR wattribute for windows that are children of the display.

**Domain** color

**Access** Modify, Reference

**Default** None

* + 1. 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

^$DI[SPLAY](display , tfacekeyword, fontface , fontsize )

where

tfacekeyword ::= expr V 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** emptystring

**Access** Reference

**Default** Not applicable

* + 1. 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** unitspec

**Access** Modify, Reference

**Default** "PIXEL"

# 

# 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.

* 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 and support build mode.

*MTERM windows* are terminal emulation windows that accept M input/output commands. The creation of elements within MTERM windows is reserved. MTERM windows do not support build mode.

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

|  |  |
| --- | --- |
| windowtype ::= expr V | APPLICATION  MTERM  implementationvalue |

* + 1. MTERM Windows

An MTERM window is opened by specifying the "MTERM" mnemonicspace in the OPEN command. If the device (i.e. the first expr in the openargument) specified in the OPEN command does not exist as a wname in ^$WINDOW, a window is created with wname equal to the device name and TYPE wattribute equal to "MTERM". If the device specified in the OPEN command exists as a wname in ^$WINDOW and that window's TYPE wattribute is equal to "MTERM", any attributes specified for that window are used. If the device specified in the OPEN command exists as a wname in ^$WINDOW and that window's TYPE wattribute is not equal to "MTERM", an error condition occurs with an ecode = "M50".

Opening an MTERM window enables it to be seen by the user provided that it is not obscured by another window or positioned outside the visible area of its display.

If an M CLOSE command designates an MTERM window that was created by the execution of an OPEN command, the window is destroyed. If an M CLOSE command designates an MTERM window that was created by the assignment of values to attributes in ^$WINDOW, the CLOSE command does not cause the window to be destroyed. If an application attempts to destroy an open MTERM window with an M KILL command, an error condition occurs with an ecode = "M51" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

* 1. Window Components

A window consists of:

An *application area*, a virtual surface on which gadgets, menus or other windows are placed. The application area is large enough to accommodate all windows, gadgets and menus placed upon it which have VISIBLE attribute values equal to TRUE.

A *window viewport* is an area in which the user can view some or all of the application area. If the application area is larger than the window viewport, it may be possible to *scroll* and/or *resize* the viewport so that a different portion of the application area is visible.

A *window frame* may surround the window viewport. The frame may contain the window title, may contain a menu bar, and may also contain controls that enable the user to resize the window, to iconify the window, and to display a system menu supplied by the underlying windowing platform.

* 1. Window Repainting

When a previously invisible window or element becomes visible or when all or part of a previously obscured window or element becomes unobscured, any necessary repainting is carried out automatically by the MWAPI implementation or by the underlying windowing platform, if any.

* 1. Window Attributes

Window attributes, identified by wattributes, define the characteristics of windows.

|  |  |
| --- | --- |
| wattribute ::= expr V | ACTIVE  BCOLOR BEVENT BLDMODE  COLOR  DEFBUTTON  DISPLAY  EVENT  FCOLOR  FFACE  FSIZE  FSTYLE GRID  ICON  ICONIFY  ID  ITITLE LINKTOPIC MAX  MENUBAR  MIN  MODAL  NEXTG  PARENT  POS PSELECT  PTYPE  RESIZE  SCROLL  SIZE  SIZEMIN  SIZEWIN  TBCOLOR  TFCOLOR  TIED  TITLE  TYPE  UNITS VALUE  VISIBLE  applicationattribute  implementationattribute |

* + 1. Applicability of Window Attributes

The applicability of wattributes to window types is specified by Table 1. A "Yes" entry indicates that the use of the attribute with the window type is defined. An empty entry indicates that the use of the attribute with the window type is reserved. implementationattribute and applicationattribute values are applicable to all window types.

| **Table 1:** **Window Attributes****Window Attributes** | | |
| --- | --- | --- |
| **Attribute** | **Application**  **Window** | **M Term**  **Window** |
| ACTIVE | Yes |  |
| BCOLOR | Yes |  |
| BEVENT | Yes |  |
| BLDMODE | Yes |  |
| COLOR | Yes |  |
| DEFBUTTON | Yes |  |
| DISPLAY | Yes | Yes |
| EVENT | Yes |  |
| FCOLOR | Yes |  |
| FFACE | Yes |  |
| FSIZE | Yes |  |
| FSTYLE | Yes |  |
| GRID | Yes |  |
| ICON | Yes |  |
| ICONIFY | Yes |  |
| ID | Yes |  |
| ITITLE | Yes |  |
| LINKTOPIC | Yes |  |
| MAX | Yes |  |
| MENUBAR | Yes |  |
| MIN | Yes |  |
| MODAL | Yes | Yes |
| NEXTG | Yes |  |
| PARENT | Yes |  |
| POS | Yes | Yes |
| PSELECT | Yes |  |
| PTYPE | Yes |  |
| RESIZE | Yes | Yes |
| SCROLL | Yes |  |
| SIZE | Yes | Yes |
| SIZEMIN | Yes |  |
| SIZEWIN | Yes |  |
| TBCOLOR | Yes |  |
| TFCOLOR | Yes |  |
| TIED | Yes |  |

|  |
| --- |
| **Table 1: Window Attributes (Continued)** |

|  |  |  |
| --- | --- | --- |
| TITLE | Yes | Yes |
| TYPE | Yes | Yes |
| UNITS | Yes | Yes |
| VALUE | Yes |  |
| VISIBLE | Yes |  |

* + 1. ACTIVE wattribute

**Definition** If ACTIVE is FALSE, the window and its descendant windows and elements are *disabled*. If a window or element is disabled, the user cannot interact with it and events do not occur for it.

If ACTIVE is TRUE for a window and all of its ancestor windows, if any, the window is not disabled.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* + 1. BCOLOR wattribute

**Definition** BCOLOR specifies the default background color for gadgets subsequently created within the window. Modifying BCOLOR does not affect gadgets that already exist.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's BCOLOR dattribute or wattribute

* + 1. BEVENT wattribute

**Definition** BEVENT specifies, via descendant nodes identified by betype subscripts, the window's callback routine(s) for the build-mode event specified by etype. The format of a window event specification is defined in Section 9.2. No value is present at the level of the BEVENT wattribute itself.

For a window that is not in build mode, the node’s descendants may be set and referenced but are ignored.

**Domain** For immediate descendant nodes

L doargument

**Access** For immediate descendant nodes

Create, Modify, Reference

**Default** None

* + 1. BLDMODE wattribute

**Definition** If BLDMODE is defined for a window, that window operates in build mode.

**Domain** emptystring

**Access** Create, Reference

**Default** None

* + 1. COLOR wattribute

**Definition** COLOR specifies the color of the window's application area.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's COLOR dattribute or wattribute

* + 1. DEFBUTTON wattribute

**Definition** DEFBUTTON specifies the default push button for the window. If the value of DEFBUTTON specifies a gadget which is a child of the window and which has a TYPE attribute value of "BUTTON", the user can "push" the specified button, that is cause it to generate a SELECT event, by pressing a key on the keyboard. The specific key to be pressed is defined by the underlying windowing platform. If focus is not on the button at the time the key is pressed, the SELECT event will be preceded by such UNFOCUS, CHANGE and FOCUS events as would have occurred if the button had been “pushed” using the pointer, during which the request to change focus may be denied.

If an application attempts to assign a value to DEFBUTTON which (a) does not specify a gadget which is a child of the window or (b) does not identify a gadget with a TYPE elattribute value equal to "BUTTON", an error condition occurs with an ecode = "M52" and $EREF containing a value that indicates the ssvn reference for which the invalid DEFBUTTON value was assigned.

The effect is not specified if DEFBUTTON identifies a gadget that has been destroyed after the value identifying it was assigned to DEFBUTTON.

If the window is in build mode the wattribute may be set and referenced but is ignored.

**Domain** gname

**Access** Create, Modify, Reference

**Default** Platform

* + 1. DISPLAY wattribute

**Definition** DISPLAY specifies the display on which the window appears.

**Domain** display

**Access** Create, Reference

**Default** Value of $PDISPLAY

* + 1. EVENT wattribute

**Definition** EVENT specifies, via descendant nodes identified by etype subscripts, the window's callback routine(s) for the event specified by etype. The format of a window event specification is defined in Section 9.2. No value is present at the level of the EVENT wattribute itself.

If the window is in build mode the EVENT wattribute descendant nodes may be set and referenced but are ignored. All events are notified via the BEVENT wattribute.

**Domain** For immediate descendant nodes

L doargument

**Access** For immediate descendant nodes

Create, Modify, Reference

**Default** None

* + 1. FCOLOR wattribute

**Definition** FCOLOR specifies the default foreground color for gadgets subsequently created within the window. Modifying FCOLOR does not affect gadgets that already exist.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's FCOLOR dattribute or wattribute

* + 1. FFACE wattribute

**Definition** FFACE specifies the default type face that is used for subsequently created gadgets in the window.

**Domain** fontface

**Access** Create, Modify, Reference

**Default** Value of parent's FFACE dattribute or wattribute

* + 1. FSIZE wattribute

**Definition** FSIZE specifies the default type size that is used for subsequently created gadgets in the window.

**Domain** fontsize

**Access** Create, Modify, Reference

**Default** Value of parent's FSIZE dattribute or wattribute

* + 1. FSTYLE wattribute

**Definition** FSTYLE specifies the default type style that is used for subsequently created gadgets in the window.

**Domain** fontstyle

**Access** Create, Modify, Reference

**Default** Value of parent's FSTYLE dattribute or wattribute

* + 1. GRID wattribute

**Definition** In conjunction with descendant nodes GRID defines the properties of a reference grid for the placement of gadgets on a build-mode window.

For a window in build mode, GRID defines the size of one rectangle in the grid. If the unit of measure is CHAR, the basis font is the font specified by the window’s FFACE, FSIZE and FSTYLE wattributes. If the unit of measure is REL, the basis size is the size most recently assigned to the SIZE wattribute. If either of the hsize or vsize components of GRID is zero, the grid dimension corresponding to that component is one pixel but no grid is visible.

Descendant nodes are of the form

^$W[INDOW] ( wname , gridkeyword , gridattr )

where

gridkeyword ::= expr V GRID

The SNAPPOS gridattr determines how gadgets behave when moved. If TRUE, when a single gadget is moved by user action that gadget’s position is adjusted to coincide with the nearest point on the grid, and when a group of gadgets is moved the origin of the group’s bounding rectangle is adjusted to coincide with the nearest point on the grid. If FALSE, no adjustment of position occurs.

The SNAPSIZE gridattr determines how a gadget behaves when resized. If TRUE, when a gadget is resized by user action edge or edges being resized are adjusted to coincide with the nearest point on the grid in the corresponding dimension. If FALSE, no adjustment of size occurs.

The VISIBLE gridattr determines visibility of the grid. If TRUE, a grid is visible on the window, so long as both dimensions of the grid’s size are positive. Appearance of the grid is platform-dependent. If FALSE, no grid is shown.

For a window that is not in build mode, the node and its descendants may be set and referenced but are ignored.

**Domain** GRID has a domain of size and its descendant nodes have a domain of truefalse.

**Access** Create, Modify, Reference

**Default** None. If the GRID node is created as a result of assigning a value to a subnode, the value of the GRID node defaults to “1,1,CHAR”. Subnodes default to TRUE.

* + 1. ICON wattribute

**Definition** ICON identifies the icon to display if the window is iconified.

**Domain** extresource

**Access** Create, Modify, Reference

**Default** Platform

* + 1. ICONIFY wattribute

**Definition** If ICONIFY is TRUE, the window contains a means (e.g. a button on the window frame) for the user to iconify the window. If ICONIFY is FALSE and if the underlying windowing platform so permits, the window does not contain a means for the user to iconify the window.

If the window is in build mode the wattribute may be set and referenced but is ignored. The window always provides a means for the user to iconify the window.

**Domain** truefalse

**Access** Create, Reference

**Default** Platform

* + 1. ID wattribute

**Definition** ID is an internal identifier for the window that assists in accessing this window in the underlying windowing platform.

**Domain** Unspecified

**Access** Reference

**Default** Not applicable

* + 1. ITITLE wattribute

**Definition** ITITLE specifies text to appear with the icon when the window is iconified. If the value of ITITLE is equal to emptystring, then no text appears.

**Domain** chars

**Access** Create, Modify, Reference

**Default** Value of the window's TITLE wattribute

* + 1. LINKTOPIC wattribute

**Definition** If defined, LINKTOPIC identifies the window as a source of data exchange to other applications

**Domain** detopic

**Access** Create

**Default** None

* + 1. MAX wattribute

**Definition** If MAX is TRUE, the window is in the maximized state. If MAX is FALSE, the window is not in the maximized state. If the application assigns a value of TRUE to MAX, the window is maximized.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** FALSE

* + 1. MENUBAR wattribute

**Definition** MENUBAR identifies the menu to be displayed in the window's menu bar. If the MENUBAR attribute is not specified when a window is created, no menu bar is created for the window. If the underlying windowing platform prevents a menu bar from being added after window creation, attempts to modify the MENUBAR attribute are ignored. If the value specified for MENUBAR is not the mname of a menu which is a child of the window, or if the value is equal to emptystring, an empty menu bar is created.

**Domain** mname or emptystring

**Access** Create, Modify, Reference

**Default** None

* + 1. MIN wattribute

**Definition** If MIN is TRUE, the window is in the iconic state. If MIN is FALSE, the window is in the open (non-iconic) state. If the application assigns a value of TRUE to MIN, the window is made iconic.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** FALSE

* + 1. MODAL wattribute

**Definition** A window for which the MODAL wattribute is defined is referred to as a *modal* window.

When a modal window is visible, other windows may be disabled. A MODAL wattribute value equal to "PARENT" indicates that the modal window disables only its parent window. A MODAL wattribute value equal to "ANCESTORS" indicates that the modal window disables all of its ancestor windows. A MODAL wattribute value equal to "APPLICATION" indicates that the modal window disables all active windows defined in ^$WINDOW except itself.

If an application attempts to create a child window of a modal window which is not also a modal window, an error condition occurs with an ecode = "M54" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If the window is in build mode the wattribute may be set and referenced but is ignored.

**Domain** modaltype

**Access** For an MTERM window: Reference

For other window types: Create, Reference

**Default** For an MTERM window: "APPLICATION"

For other window types: None

* + 1. NEXTG wattribute

**Definition** NEXTG indicates the action that occurs if this window receives focus but no gadget within the window receives focus. NEXTG only has effect the first time a window receives focus after it is created or the first time a window receives focus after its VISIBLE wattribute value changes from FALSE to TRUE.

If an application attempts to assign a value to NEXTG that does not identify a gadget that has been defined for the window, an error condition occurs with an ecode = "M49" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If the NEXTG wattribute for window W identifies a gadget G that cannot get focus, that gadget's NEXTG elattribute determines where focus goes. This process is repeated for successive Gs until (a) gadget G's NEXTG elattribute identifies a previously identified gadget G or gadget G's NEXTG elattribute is not specified, in which case focus remains at window W; or (b) gadget G's NEXTG elattribute identifies a gadget Gn which can get focus is reached, in which case focus goes to gadget Gn.

The effect is not specified if NEXTG identifies a gadget that has been destroyed after the value identifying it was assigned to NEXTG.

If the window is in build mode the wattribute may be set and referenced but is ignored.

**Domain** gname

**Access** Create, Modify, Reference

**Default** None

* + 1. PARENT wattribute

**Definition** PARENT identifies the window that is this window's parent. If a window has a parent window, it (a) becomes invisible when its parent becomes invisible or is iconified; (b) if its VISIBLE attribute is TRUE, becomes visible when its parent becomes visible and is not iconified; and (c) is destroyed if its parent is destroyed.

If a window's parent is a display, the PARENT wattribute is not defined.

If, for a window A, an application attempts to assign a value to PARENT which identifies a window B, and the DISPLAY wattribute values for windows A and B are not equal, an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

The effect of assigning a value for the PARENT wattribute which causes a single window to be both an ancestor and a descendant of another window is not specified.

**Domain** wname

**Access** Create, Reference

**Default** None

* + 1. POS wattribute

**Definition** POS specifies the position of the origin of the window frame, expressed in the unit of measure defined by the unitspec component of the pos value, if present, or by the window's UNITS wattribute otherwise.

If the window's TIED wattribute is TRUE, the position is relative to the origin of the viewport of the window's parent. If the unit of measure is CHAR, the basis font is the font specified by the parent's FFACE, FSIZE and FSTYLE wattributes. If the unit of measure is REL, the basis size is the size of the parent's viewport when the parent is created.

If the window's TIED wattribute is FALSE, the position is relative to the origin of the display. If the unit of measure is CHAR, the basis font is the font specified by the FFACE, FSIZE, and FSTYLE dattributes for the display on which the window appears. If the unit of measure is REL, the basis size is the size of the display.

When an application assigns a value to the POS wattribute of an existing window, if hpos or vpos is not specified then the horizontal or vertical position of the window, respectively, is not altered.

For a window that is iconic, the effect of assigning a value to POS is platform dependent.

**Domain** pos

**Access** Create, Modify, Reference

**Default** If hpos, vpos, or both are not specified when a window is created, defaults for the unspecified values are Platform.

* + 1. PSELECT wattribute

**Definition** For a build-mode window PSELECT specifies the gadget which is the principal selection. When more than one gadget is selected they may be moved as a group but only one, the principaly selection, may be resized. If PSELECT is undefined, no gadget is selected.

If an application assigns a value to PSELECT which identifies a gadget on the window, that gadget becomes the principal selection and is added as a descendant to the VALUE wattribute if necessary. If an application assigns a value to PSELECT which does not identify a gadget which is a child of the window, an error condition occurs with $ECODE containing”,M48,” and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If an application kills the PSELECT wattribute it becomes undefined and the VALUE wattribute is also implicitly killed.

For a window that is not in build mode, the wattribute may be set and referenced but is ignored.

**Domain** gname

**Access** Create, Modify, Reference

**Default** None

* + 1. PTYPE wattribute

**Definition** PTYPE specifies the pointer appearance when the pointer is in the window viewport.

If the display on which the window appears does not have a pointer, any value of PTYPE assigned by an application is ignored.

**Domain** ptype

**Access** Create, Modify, Reference

**Default** None

* + 1. RESIZE wattribute

**Definition** If RESIZE is TRUE, the user can alter the size of the window. If RESIZE is FALSE, the user cannot alter the size of the window.

If the window is in build mode the wattribute may be set and referenced but is ignored. The user can always alter the size of the window.

**Domain** truefalse

**Access** Create, Reference

**Default** Platform

* + 1. SCROLL wattribute

**Definition** If SCROLL is TRUE and a portion of the application area lies outside the window viewport, a horizontal and/or a vertical scroll bar is present, enabling the user to scroll the application area. Otherwise, scroll bars for scrolling the application area are not present, if the underlying windowing platform permits them to be omitted.

**Domain** truefalse

**Access** Create, Reference

**Default** TRUE

* + 1. SIZE wattribute

**Definition** SIZE specifies the width and height of the window viewport, expressed in the unit of measure defined by the unitspec component of the size value, if present, and by the window's UNITS wattribute otherwise.

If the unit of measure is CHAR, the basis font is the font specified by the window's FFACE, FSIZE and FSTYLE wattributes.

If the window's TIED wattribute is TRUE and the unit of measure is REL, the basis size is the size of the parent's window viewport when the parent is created. If the window's TIED wattribute is FALSE and the unit of measure is REL, the basis size is the size of the display.

When an application assigns a value to the SIZE wattribute of an existing window, if hsize or vsize is not specified then the horizontal or vertical size of the window, respectively, is not altered.

If the window is in build mode, setting the SIZE re-establishes the basis size used for subsequent interpretation of REL units. Resizing of the viewport as a consequence of user action does not alter the basis size.

**Domain** size

**Access** Create, Modify, Reference

**Default** If hsize, vsize, or both are not specified when a window is created, defaults for the unspecified values are determined as described below.

When an application window is created, the MWAPI attempts to make it large enough to encompass all visible gadgets, defined at that time, whose POS elattributes do not contain negative values for hpos or vpos. However, if the position or size of any such gadget is defined with a unit of measure of REL, the window hsize, vsize, or both are equal to seventy-five percent of the size of the equivalent dimension of the display on which the window appears. Subsequent creation, destruction or movement of gadgets does not cause the SIZE value to change.

When an MTERM window is created, an implementation may determine a default size based on openarguments specified for the window. If the default size is not determined in this way, the default size is the equivalent of "80,24,CHAR".

* + 1. SIZEMIN wattribute

**Definition** SIZEMIN specifies the minimum size to which the user can shrink the window viewport, expressed in the unit of measure defined by the unitspec component of the size value, if present, and by the window's UNITS wattribute otherwise. Any attempt by the user to make the window viewport smaller than SIZEMIN is ignored.

If the unit of measure is CHAR, the basis font is the font specified by the window's FFACE, FSIZE and FSTYLE wattributes.

When an application assigns a value to the SIZEMIN wattribute of an existing window, if hsize or vsize is not specified then the minimum horizontal or vertical size of the window, respectively, is not altered.

**Domain** size

**Access** Create, Modify, Reference

**Default** Platform

* + 1. SIZEWIN wattribute

**Definition** SIZEWIN specifies the overall size of the window (including the window frame, if any) expressed in the unit of measure defined by the unitspec component of the size value, if present, and by the window's UNITS wattribute otherwise. If the underlying windowing platforms does not provide this size exactly, a best estimate is provided.

If an application attempts to assign, with a single M MERGE command, values for both the SIZE and SIZEWIN wattributes, it is not specified which of these wattributes determines the window size.

If the unit of measure is CHAR, the basis font is the font specified by the window's FFACE, FSIZE and FSTYLE wattributes.

When an application assigns a value to the SIZEWIN wattribute of an existing window, if hsize or vsize is not specified then the horizontal or vertical size of the window, respectively, is not altered.

**Domain** size

**Access** Create, Modify, Reference

**Default** Not applicable. If a value is not assigned to SIZEWIN, its value is determined implicitly by the value of SIZE.

* + 1. TBCOLOR wattribute

**Definition** TBCOLOR specifies the default title background color for gadgets subsequently created within the window. Modifying TBCOLOR does not affect gadgets that already exist.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's TBCOLOR dattribute or wattribute if any, otherwise none.

* + 1. TFCOLOR wattribute

**Definition** TFCOLOR specifies the default title foreground color for gadgets subsequently created within the window. Modifying TFCOLOR does not affect gadgets that already exist.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's TFCOLOR dattribute or wattribute if any, otherwise none.

* + 1. TIED wattribute

**Definition** If TIED is TRUE (a) if this window and its parent overlap, this window is always on top of its parent; and (b) this window maintains its position, relative to its parent's origin, when its parent moves.

If an application attempts to assign a value to TIED which is equal to TRUE and the window's parent is a display, an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** truefalse

**Access** Create, Reference

**Default** FALSE

* + 1. TITLE wattribute

**Definition** TITLE specifies the window title text. If the TITLE attribute is not specified when a window is created, there may be no title area in the window border. If the underlying windowing platform prevents a title from being added or removed after window creation, attempts to modify the TITLE attribute are ignored. If the value specified for TITLE is an empty string, an empty title area is created.

The presence or absence of a TITLE value may be ignored if the underlying windowing platform does not permit or requires a title to be displayed for a window as a result of the values assigned to other wattributes.

If the window is in build mode the wattribute may be set and referenced but is ignored. A title area is always present.

**Domain** chars

**Access** Create, Modify, Reference

**Default** For an application window, none. For an MTERM window, the default is the value of the wname that identifies the window.

* + 1. TYPE wattribute

**Definition** TYPE specifies the window type.

**Domain** windowtype

**Access** Create, Reference

**Default** For windows implicitly created by an M OPEN command , "MTERM".

For other windows, "APPLICATION".

* + 1. UNITS wattribute

**Definition** UNITS indicates the unit of measure used to specify the size and position of the window.

**Domain** unitspec

**Access** Create, Modify, Reference

**Default** For an application window:

Value of parent's UNITS dattribute or wattribute

For an MTERM window:

"CHAR", with a basis font defined by an FFACE value of "M,FIXED", an FSIZE value of "12", and an FSTYLE value of "NORMAL".

* + 1. VALUE wattribute

**Definition** For a build-mode window VALUE and its descendant nodes specify the currently selected gadget(s). VALUE specifies the number of gadgets selected, and selections are indicated by descendant nodes of the form

^$W[INDOW] (wname , vkeyword , gname )

where

vkeyword ::= expr V VALUE

and gname identifies a selected gadget.

If an application attempts to create a descendant node of VALUE which does not identify a gadget which is a child of the window, an error condition occurs with $ECODE containing”,M48,” and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If an application kills the descendant node of VALUE which identifies the principal gadget, the PSELECT wattribute is reset to be the first remaining descendant node of VALUE, if any, otherwise becoming undefined.

If an application kills the VALUE wattribute it is reset to its default and the PSELECT wattribute is also implicitly killed.

For a window that is not in build mode, the wattribute and its descendant nodes may be set and referenced but are ignored.

**Domain** VALUE has a domain of intexpr and descendant nodes have a domain of emptystring.

**Access** Create, Modify, Reference

**Default** 0

* + 1. VISIBLE wattribute

**Definition** If VISIBLE is FALSE, a window cannot be seen by the user. If VISIBLE is TRUE, a window can be seen by the user provided that it is not obscured and it is not a child of an invisible window.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

# 

# Elements

Elements perform specialized tasks within windows. Elements include gadgets, menus, and timers.

* 1. Gadgets

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

^$W[INDOW] ( wname , gkeyword , gname , elattribute )

gname is the identifier for a gadget and wname is the identifier for the window that is the parent of that gadget.

gkeyword ::= expr V G

The MWAPI defines the following gadget types, specified via gtype value for the TYPE elattribute.

|  |  |
| --- | --- |
| gtype ::= expr V | BUTTON  CHECK  DOCUMENT  FRAME  GENERIC  LABEL  LIST  LISTBUTTON  LISTENTRY  LONGLIST  RADIO  SCROLL  SYMBOL  TABLE  TEXT  implementationvalue |

* + 1. Check Box

A *check box* gadget enables the user or the application to turn an indicator on or off. Typically, a check box appears as text accompanied by an area that can be selected or deselected. A check box has a TYPE elattribute value equal to "CHECK".

* + 1. Document

A *document* gadget enables the user to view, enter, and modify one or more lines of text. Text editing capabilities are provided by the underlying windowing platform or the MWAPI. If the text cannot all be simultaneously displayed in the gadget, horizontal and vertical scrolling mechanisms can be specified. A document has a TYPE elattribute value equal to "DOCUMENT".

* + 1. Generic Box

A *generic box* is an area in which an application can draw text, geometric figures, and other graphical objects and can receive events. A generic box has a TYPE elattribute value equal to "GENERIC".

* + 1. Group Frame

A *group frame* is a rectangular outline used to visually group gadgets on a window. A group frame can also be a vertical or horizontal line. A group frame cannot get focus and events do not occur for it. A group frame has a TYPE elattribute value equal to "FRAME".

* + 1. Label

A *label* is static text that appears within a window. A label cannot get focus and events do not occur for it. A label has a TYPE elattribute value equal to "LABEL".

* + 1. List Box

A *list box* displays a list of items and enables the user to select one or more of them. If the list box is not large enough to show all items at once, a scrolling mechanism enables the user to move through the list. Such navigation through the list is handled automatically by the underlying windowing platform and the MWAPI. A list box has a TYPE elattribute value equal to "LIST".

* + 1. List Button

A *list button* is a combination of a push button and a list. Normally, the list is invisible. When the push button is pushed, the list becomes visible. When the user selects an item from the list, the list becomes invisible. The currently selected value is displayed with the push button. A list button has a TYPE elattribute value equal to "LISTBUTTON".

* + 1. List Entry Box

A *list entry box* is a combination of a text entry area (similar to a text gadget) and a list area (similar to a list box gadget.) It enables a user to select an item from the list area, which is then displayed in the text entry area. The user can enter or modify text in the text entry area. The resulting value need not be present in the list area. A list entry box has a TYPE elattribute value equal to "LISTENTRY".

* + 1. Long List Box

Like a list box, a *long list box* displays a list of items and enables the user to select one or more of them. Unlike a list box, however, navigation through the list is handled by the application. The long list box is intended for use with lists that are very large, enabling an application to optimize its construction of the list in response to user actions. A long list box has a TYPE elattribute value equal to "LONGLIST".

* + 1. Push Button

A *push button* is a gadget that the user can "press", typically to cause an event to occur. A push button has a TYPE elattribute value equal to "BUTTON".

* + 1. Radio Button Set

A *radio button set* is a collection of related items that the user can select. At any time, at most one item can be selected. Selecting an item causes any previously selected item to be deselected. A radio button set has a TYPE elattribute value equal to "RADIO".

* + 1. Scroll

A *scroll* gadget visually represents a value on a numeric scale through some sort of position indicator. The user can modify the value by moving the position indicator. A scroll gadget does not directly cause anything to scroll, although applications may employ scroll gadgets to do so. A scroll gadget can be horizontally or vertically oriented. A scroll gadget has a TYPE elattribute value equal to "SCROLL".

* + 1. Symbol

A *symbol* is an image to be displayed within a window. Symbol gadgets cannot get focus and events do not occur for them. A symbol has a TYPE elattribute value equal to "SYMBOL".

* + 1. Table

A *table* gadget enables a user to view and select choices which are displayed in rows and columns. The intersection of a row and a column is called a *cell*. Selected cells are visually highlighted in some manner. Support of non-contiguous multi-cell selection may or may not be provided by an implementation.

If all cells cannot be simultaneously displayed in the gadget, the underlying windowing platform or the MWAPI provides a scrolling mechanism. At any time, at most one row (the bottom most) and at most one column (the right most) are partially displayed. All other rows and columns are displayed either fully or not at all.

A cell can display text, an image or both. A text value assigned to a cell is displayed to the maximum extent permitted by the dimensions of the cell. Multiple lines of text are displayed in a cell if the row height is sufficient and if either (a) the text will not fit all on one line or (b) the text contains embedded linend characters.

An application can dynamically insert or delete rows and columns and change cell values.

A table gadget has a TYPE elattribute value equal to "TABLE".

* + 1. Text

A *text* gadget enables the user to view, enter, and modify a single line of text. Text editing capabilities are provided by the underlying windowing platform or the MWAPI. If the text cannot all be simultaneously displayed in the gadget, horizontal scrolling mechanisms are provided. A text gadget has a TYPE elattribute value equal to "TEXT".

* 1. Menus

A *menu* is a list of choices from which a user makes a selection, typically to cause an action to occur immediately. The MWAPI defines two types of menus: a *menu bar* which typically displays choices horizontally and appears at the top of a window or display; and a *pop-up menu* which typically displays choices vertically. A pop-up menu can appear descendant from a menu bar or somewhere else in the window, completely independent of the menu bar.

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

^$W[INDOW] ( wname , mkeyword , mname , elattribute )

mname is the identifier for a menu and wname is the identifier for the window that is the parent of that menu.

mkeyword ::= expr V M

* 1. Timers

A *timer* generates an event after a specified time interval elapses. It has no visible representation.

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

^$W[INDOW] ( wname , tkeyword , tname , elattribute )

tname is the identifier for a timer and wname is the identifier for the window that is the parent of that timer.

tkeyword ::= expr V T

Provided that event processing is activated and that the ACTIVE wattribute has a value of TRUE for all ancestor windows and that the timer's INTERVAL elattribute has a value and that timer events are enabled for a timer, an interval timer starts (a) whenever a value is assigned to one or more of the timer's elattributes or (b) whenever a value of TRUE is assigned to the ACTIVE wattribute of an ancestor window or (c) the application executes an explicit or implicit M QUIT command from a callback routine processing a timer event for the timer.

An interval timer is cancelled (a) whenever a value is assigned to or removed from one or more of the timer's elattributes or (b) the timer is destroyed or (c) whenever a value of FALSE is assigned to the ACTIVE wattribute of an ancestor window.

If, after a timer starts, the time period specified by the timer's INTERVAL elattribute elapses without the interval timer being cancelled, the interval timer expires and a timer event occurs for the timer.

## Element Attributes

Element attributes, identified by elattributes, define the characteristics of elements.

|  |  |
| --- | --- |
| elattribute ::= expr V | ACTIVE  BCOLOR BLDMODE  CANCEL  CANCHANGE  CHANGED  CHARMAX  CHOICE  COL  COLWIDTH  DRAW  DRAWTYPE  EVENT  FCOLOR  FFACE  FRAMED  FSIZE  FSTYLE  GRID  HCOLID  HROWID  ID  INSELECT  INTERVAL  LEFTSHOW LINKACTION LINKITEM LINKMODE LINKSOURCE LINKTIMEOUT  NEXTG  POS  PTYPE  RESOURCE  ROW  ROWCOL |

|  |  |
| --- | --- |
|  | ROWHEIGHT  SCROLL  SCROLLBY  SCROLLDIR  SCROLLPOS  SCROLLRANGE  SELECTMAX  SELECTVAL  SIZE  TBCOLOR  TFCOLOR  TFFACE  TFSIZE  TFSTYLE  TITLE  TOPSHOW  TPOS  TYPE  UNITS  VALUE  VISIBLE  applicationattribute  implementationattribute |

* + 1. Applicability of Element Attributes

The applicability of elattributes to element types is specified by Table 2. A "Yes" entry indicates that the use of the attribute with the element type is defined. An empty entry indicates that the use of the attribute with the element type is reserved. implementationattribute and applicationattribute values are applicable to all element types.

| **Table 2: Element Attributes****Element Attributes** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | Check  Box | Docu-ment | Generic  Box | Group  Frame | Label | List  Box |
| ACTIVE | Yes | Yes | Yes |  |  | Yes |
| BCOLOR | Yes | Yes | Yes | Yes | Yes | Yes |
| BLDMODE | Yes | Yes | Yes | Yes | Yes | Yes |
| CANCEL | Yes | Yes | Yes |  |  | Yes |
| CANCHANGE | Yes | Yes |  |  |  | Yes |
| CHANGED | Yes | Yes |  |  |  | Yes |
| CHARMAX |  | Yes |  |  |  |  |
| CHOICE |  |  |  |  |  | Yes |
| COL |  |  |  |  |  |  |
| COLWIDTH |  |  |  |  |  |  |
| DRAW |  |  | Yes |  |  |  |
| DRAWTYPE |  |  | Yes |  |  |  |
| EVENT | Yes | Yes | Yes |  |  | Yes |
| FCOLOR | Yes | Yes | Yes | Yes | Yes | Yes |
| FFACE |  | Yes | Yes |  |  | Yes |
| FRAMED |  | Yes | Yes |  | Yes |  |
| FSIZE |  | Yes | Yes |  |  | Yes |
| FSTYLE |  | Yes | Yes |  |  | Yes |
| GRID |  |  |  |  |  |  |
| HCOLID |  |  |  |  |  |  |
| HROWID |  |  |  |  |  |  |
| ID | Yes | Yes | Yes | Yes | Yes | Yes |
| INSELECT |  | Yes |  |  |  |  |
| INTERVAL |  |  |  |  |  |  |
| LEFTSHOW |  |  |  |  |  |  |
| LINKACTION |  | Yes |  |  |  |  |
| LINKITEM |  | Yes |  |  |  |  |
| LINKMODE |  | Yes |  |  |  |  |
| LINKSOURCE |  | Yes |  |  |  |  |
| LINKTIMEOUT |  | Yes |  |  |  |  |
| NEXTG | Yes | Yes | Yes | Yes | Yes | Yes |

| **Table 2: Element Attributes (Continued)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | Check  Box | Docu-ment | Generic  Box | Group  Frame | Label | List  Box |
| POS | Yes | Yes | Yes | Yes | Yes | Yes |
| PTYPE | Yes | Yes | Yes | Yes | Yes | Yes |
| RESOURCE |  |  |  |  |  |  |
| ROW |  |  |  |  |  |  |
| ROWCOL |  |  |  |  |  |  |
| ROWHEIGHT |  |  |  |  |  |  |
| SCROLL |  | Yes |  |  |  |  |
| SCROLLBY |  |  |  |  |  |  |
| SCROLLDIR |  |  |  |  |  |  |
| SCROLLPOS |  |  |  |  |  |  |
| SCROLLRANGE |  |  |  |  |  |  |
| SELECTMAX |  |  |  |  |  | Yes |
| SELECTVAL |  | Yes |  |  |  |  |
| SIZE | Yes | Yes | Yes | Yes | Yes | Yes |
| TBCOLOR |  | Yes |  |  |  | Yes |
| TFCOLOR |  | Yes |  |  |  | Yes |
| TFFACE | Yes | Yes |  | Yes | Yes | Yes |
| TFSIZE | Yes | Yes |  | Yes | Yes | Yes |
| TFSTYLE | Yes | Yes |  | Yes | Yes | Yes |
| TITLE | Yes | Yes |  | Yes | Yes | Yes |
| TOPSHOW |  |  |  |  |  | Yes |
| TPOS |  | Yes |  |  |  | Yes |
| TYPE | Yes | Yes | Yes | Yes | Yes | Yes |
| UNITS | Yes | Yes | Yes | Yes | Yes | Yes |
| VALUE | Yes | Yes |  |  |  | Yes |
| VISIBLE | Yes | Yes | Yes | Yes | Yes | Yes |

| **Table 2: Element Attributes (Continued)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | List  Button | List  Entry  Box | Long  List  Box | Menu | Push  Button | Radio  Button  Set |
| ACTIVE | Yes | Yes | Yes | Yes | Yes | Yes |
| BCOLOR | Yes | Yes | Yes |  |  | Yes |
| BLDMODE | Yes | Yes | Yes |  | Yes | Yes |
| CANCEL | Yes | Yes | Yes |  | Yes | Yes |
| CANCHANGE | Yes | Yes | Yes |  |  | Yes |
| CHANGED | Yes | Yes | Yes |  |  | Yes |
| CHARMAX |  | Yes |  |  |  |  |
| CHOICE | Yes | Yes | Yes | Yes |  | Yes |
| COL |  |  |  |  |  |  |
| COLWIDTH |  |  |  |  |  |  |
| DRAW |  |  |  |  |  |  |
| DRAWTYPE |  |  |  |  |  |  |
| EVENT | Yes | Yes | Yes |  | Yes | Yes |
| FCOLOR | Yes | Yes | Yes |  |  | Yes |
| FFACE | Yes | Yes | Yes |  |  | Yes |
| FRAMED |  |  |  |  |  | Yes |
| FSIZE | Yes | Yes | Yes |  |  | Yes |
| FSTYLE | Yes | Yes | Yes |  |  | Yes |
| GRID |  |  |  |  |  |  |
| HCOLID |  |  |  |  |  |  |
| HROWID |  |  |  |  |  |  |
| ID | Yes | Yes | Yes | Yes | Yes | Yes |
| INSELECT |  | Yes |  |  |  |  |
| INTERVAL |  |  |  |  |  |  |
| LEFTSHOW |  |  |  |  |  |  |
| LINKACTION |  |  |  |  |  |  |
| LINKITEM |  |  |  |  |  |  |
| LINKMODE |  |  |  |  |  |  |
| LINKSOURCE |  |  |  |  |  |  |
| LINKTIMEOUT |  |  |  |  |  |  |
| NEXTG | Yes | Yes | Yes |  | Yes | Yes |

| **Table 2: Element Attributes (Continued)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | List  Button | List  Entry  Box | Long  List  Box | Menu | Push  Button | Radio  Button  Set |
| POS | Yes | Yes | Yes | Yes | Yes | Yes |
| PTYPE | Yes | Yes | Yes |  | Yes | Yes |
| RESOURCE |  |  |  |  | Yes |  |
| ROW |  |  |  |  |  |  |
| ROWCOL |  |  |  |  |  | Yes |
| ROWHEIGHT |  |  |  |  |  |  |
| SCROLL |  |  |  |  |  |  |
| SCROLLBY |  |  |  |  |  |  |
| SCROLLDIR |  |  |  |  |  |  |
| SCROLLPOS |  |  | Yes |  |  |  |
| SCROLLRANGE |  |  | Yes |  |  |  |
| SELECTMAX |  |  | Yes |  |  |  |
| SELECTVAL |  | Yes |  |  |  |  |
| SIZE | Yes | Yes | Yes |  | Yes | Yes |
| TBCOLOR | Yes | Yes | Yes |  |  | Yes |
| TFCOLOR | Yes | Yes | Yes |  |  | Yes |
| TFFACE | Yes | Yes | Yes |  | Yes | Yes |
| TFSIZE | Yes | Yes | Yes |  | Yes | Yes |
| TFSTYLE | Yes | Yes | Yes |  | Yes | Yes |
| TITLE | Yes | Yes | Yes |  | Yes | Yes |
| TOPSHOW |  | Yes | Yes |  |  |  |
| TPOS | Yes | Yes | Yes |  |  | Yes |
| TYPE | Yes | Yes | Yes |  | Yes | Yes |
| UNITS | Yes | Yes | Yes | Yes | Yes | Yes |
| VALUE | Yes | Yes | Yes |  |  | Yes |
| VISIBLE | Yes | Yes | Yes | Yes | Yes | Yes |

| **Table 2: Element Attributes (Continued)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | Scroll | Symbol | Table | Text | Timer |
| ACTIVE | Yes |  | Yes | Yes | Yes |
| BCOLOR | Yes |  | Yes | Yes |  |
| BLDMODE | Yes | Yes | Yes | Yes |  |
| CANCEL |  |  | Yes | Yes |  |
| CANCHANGE | Yes |  | Yes | Yes |  |
| CHANGED | Yes |  | Yes | Yes |  |
| CHARMAX |  |  |  | Yes |  |
| CHOICE |  |  | Yes |  |  |
| COL |  |  | Yes |  |  |
| COLWIDTH |  |  | Yes |  |  |
| DRAW |  |  |  |  |  |
| DRAWTYPE |  |  |  |  |  |
| EVENT | Yes |  | Yes | Yes | Yes |
| FCOLOR | Yes |  | Yes | Yes |  |
| FFACE |  |  | Yes | Yes |  |
| FRAMED |  |  |  | Yes |  |
| FSIZE |  |  | Yes | Yes |  |
| FSTYLE |  |  | Yes | Yes |  |
| GRID |  |  | Yes |  |  |
| HCOLID |  |  | Yes |  |  |
| HROWID |  |  | Yes |  |  |
| ID | Yes | Yes | Yes | Yes | Yes |
| INSELECT |  |  |  | Yes |  |
| INTERVAL |  |  |  |  | Yes |
| LEFTSHOW |  |  | Yes |  |  |
| LINKACTION |  |  |  | Yes |  |
| LINKITEM |  |  |  | Yes |  |
| LINKMODE |  |  |  | Yes |  |
| LINKSOURCE |  |  |  | Yes |  |
| LINKTIMEOUT |  |  |  | Yes |  |
| NEXTG | Yes | Yes | Yes | Yes |  |

| **Table 2: Element Attributes (Continued)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | Scroll | Symbol | Table | Text | Timer |
| POS | Yes | Yes | Yes | Yes |  |
| PTYPE | Yes | Yes | Yes | Yes |  |
| RESOURCE |  | Yes |  |  |  |
| ROW |  |  | Yes |  |  |
| ROWCOL |  |  |  |  |  |
| ROWHEIGHT |  |  | Yes |  |  |
| SCROLL |  |  |  |  |  |
| SCROLLBY | Yes |  |  |  |  |
| SCROLLDIR | Yes |  |  |  |  |
| SCROLLPOS |  |  |  |  |  |
| SCROLLRANGE | Yes |  |  |  |  |
| SELECTMAX |  |  |  |  |  |
| SELECTVAL |  |  |  | Yes |  |
| SIZE | Yes | Yes | Yes | Yes |  |
| TBCOLOR | Yes |  | Yes | Yes |  |
| TFCOLOR | Yes |  | Yes | Yes |  |
| TFFACE | Yes |  | Yes | Yes |  |
| TFSIZE | Yes |  | Yes | Yes |  |
| TFSTYLE | Yes |  | Yes | Yes |  |
| TITLE | Yes |  | Yes | Yes |  |
| TOPSHOW |  |  | Yes |  |  |
| TPOS | Yes |  | Yes | Yes |  |
| TYPE | Yes | Yes | Yes | Yes |  |
| UNITS | Yes | Yes | Yes | Yes |  |
| VALUE | Yes |  | Yes | Yes |  |
| VISIBLE | Yes | Yes | Yes | Yes |  |

* + 1. ACTIVE elattribute

**Definition** If ACTIVE is FALSE, the element is disabled; the user cannot interact with the element and events do not occur for it.

If ACTIVE is TRUE, the element is enabled.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* + 1. BCOLOR elattribute

**Definition** BCOLOR specifies the background color for the gadget.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's BCOLOR wattribute

* + 1. BLDMODE elattribute

**Definition** For a build-mode window BLDMODE specifies the actions applicable to the gadget. If the value of BLDMODE includes MOVE, the gadget can be moved within the parent window’s viewport by user action. If the value of BLDMODE includes SIZE, the gadget can be resized within the parent window’s viewport by user action. If the value of BLDMODE is emptystring, the gadget cannot be selected by user action and attempts to specify it as a subnode of the parent window’s VALUE wattribute are ingnored.

For a window that is not in build mode, the elattribute may be set and referenced but is ignored.

**Domain** buildops

**Access** Create, Reference

**Default** For a build-mode window: “MOVE,SIZE”

For a window not in build mode: None

* + 1. CANCEL elattribute

**Definition** If CANCEL is TRUE for an element then, when the element receives focus, any change or unfocus events that would have occurred for the gadget that previously had focus do not occur.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** FALSE

* + 1. CANCHANGE elattribute

**Definition** If CANCHANGE is TRUE, user actions can cause the VALUE attribute for the gadget to change. If CANCHANGE is FALSE, user actions cannot cause the VALUE attribute for the gadget to change, although some user interaction with the gadget may be possible.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* + 1. CHANGED elattribute

**Definition** CHANGED is set to TRUE by the MWAPI implementation whenever the user causes the VALUE elattribute for the gadget to change, even if the net result of a series of modifications leaves the VALUE elattribute effectively unchanged.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** FALSE

* + 1. CHARMAX elattribute

**Definition** CHARMAX specifies the maximum number of characters the text value of a gadget can have as a result of user entry or modification of text. Embedded linend characters are included in the count. The maximum character limit may not be enforced for modifications of the text value by the application. A value of 0 (zero) indicates that there is no explicit limit on the text value length.

**Domain** intexpr

**Access** Create, Reference

**Default** 0 (zero)

* + 1. CHOICE elattribute

**Definition** CHOICE specifies, via descendant nodes identified by item subscripts, the text to be displayed for choices. The sequence of choices is determined by the collating sequence of their item, ritem, and citem subscripts, beginning at the edge closest to the origin.

For a table gadget, the value of an entry of the form

^$W[INDOW]( wname , gkeyword , gname , ckeyword , ritem , citem)

specifies the text that appears for the cell identified by the ritem and citem subscripts for the gadget identified by gname.

ckeyword ::= expr V CHOICE

The ritem subscript identifies a row. If there is no such row, it is created.

The citem subscript identifies a column. If there is no such column, it is created.

For gadget types other than table, the value of an entry of the form

^$W[INDOW]( wname , gkeyword , gname , ckeyword , item )

specifies the text that appears for the choice identified by item for the gadget identified by gname.

For a menu, the value of an entry of the form

^$W[INDOW]( wname , mkeyword , mname , ckeyword , item )

specifies the text that appears for the choice identified by item for the menu identified by mname.

No value is present at the level of the CHOICE elattribute itself.

For a menu or radio button set, if the value of a descendant node of the CHOICE elattribute contains a mnemonichar, the character following the mnemonichar is a mnemonic character that may enable the user, via the keyboard, to select the choice. The mnemonichar is not displayed. If two mnemonichars are immediately adjacent within a CHOICE value, no mnemonic character is defined and a single mnemonichar appears in the displayed choice text. The character mnemonichar cannot itself be used as a mnemonic character.

**Domain** chars

**Access** Create, Modify, Reference

**Default** For a table gadget, none.

For other element types, when no value is assigned at the item level but one or more descendant nodes is present, the default at the item level is

item

* + 1. COL elattribute

**Definition** COL specifies the columns in a table gadget, via descendant nodes of the form

^$W[INDOW]( wname , gkeyword , gname , colkeyword , citem )

where

colkeyword ::= expr V COL

No value is present at the level of the COL elattribute itself. There is a value of emptystring at the citem level.

A COL elattribute descendant is present for every citem subscript that is present as a descendant of the CHOICE elattribute for the gadget. If an application assigns a CHOICE descendant with a citem subscript that is not present as a COL descendant, a COL descendant with that citem subscript is automatically created.

A COL elattribute descendant may be present with a citem subscript which is not present as a descendant of the CHOICE elattribute for the gadget. Such a citem subscript defines an empty column.

**Domain** emptystring

**Access** Create, Modify, Reference

**Default** Not Applicable.

* + 1. COLWIDTH elattribute

**Definition** COLWIDTH specifies the default width for columns in a gadget.

If the unit of measure is CHAR, the basis font is the font specified by the gadget's FFACE, FSIZE and FSTYLE elattributes. If the unit of measure is REL, the basis size is the size of the gadget at creation.

**Domain** width

**Access** Create, Modify, Reference

**Default** Platform.

* + 1. DRAW elattribute

**Definition** DRAW specifies the number of drawcommands that are descendants of the DRAW elattribute of the gadget. An application cannot assign a value to DRAW. An application can use the M KILL command to remove the DRAW elattribute and its descendant drawcommands. This removes the drawing specified by the drawcommands and the DRAW elattribute is set to 0 (zero).

The format of drawcommands which are descendant to DRAW is specified in Section 7.

**Domain** intlit

**Access** Modify, Reference

**Default** 0 (zero)

* + 1. DRAWTYPE elattribute

**Definition** DRAWTYPE specifies the type of drawcommands that can be specified in the DRAW attribute of the gadget.

**Domain** drawtype

**Access** Create, Reference

**Default** MDRAW

* + 1. EVENT elattribute

**Definition** EVENT specifies, via descendant nodes identified by etype subscripts, the element's callback routine(s) for the event specified by etype. The format of an element event specification is defined in Section 9.2. No value is present at the level of the EVENT elattribute itself.

If an element’s parent window is in build mode the EVENT elattribute descendant nodes may be set and referenced but are ignored. No element events occur.

**Domain** For immediate descendants

L doargument

**Access** For immediate descendants

Create, Modify, Reference

**Default** None

* + 1. FCOLOR elattribute

**Definition** FCOLOR specifies the foreground color for the gadget. For gadgets containing text, FCOLOR specifies the color of the text.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent's FCOLOR wattribute

* + 1. FFACE elattribute

**Definition** FFACE specifies the type face used for all text that appears within the gadget except the title.

**Domain** fontface

**Access** Create, Reference

**Default** Value of parent window's FFACE wattribute

* + 1. FRAMED elattribute

**Definition** If FRAMED is TRUE, a frame appears at the perimeter of the gadget.

**Domain** truefalse

**Access** Create, Reference

**Default** For label gadgets, FALSE. For other gadget types, TRUE.

* + 1. FSIZE elattribute

**Definition** FSIZE specifies the type size used for all text that appears within the gadget except the title.

**Domain** fontsize

**Access** Create, Reference

**Default** Value of parent window's FSIZE wattribute

* + 1. FSTYLE elattribute

**Definition** FSTYLE specifies the type style used for all text that appears within the gadget except the title.

**Domain** fontstyle

**Access** Create, Reference

**Default** Value of parent window's FSTYLE wattribute

* + 1. GRID elattribute

**Definition** If GRID is TRUE for a table gadget, grid lines are displayed between the rows and columns of the table. If GRID is FALSE, grid lines are not displayed.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* + 1. HCOLID elattribute

**Definition** In a table gadget, any column whose citem does not follow the value of HCOLID (in the collating sequence of ^$WINDOW) is a header column. Header columns do not scroll and are displayed in a manner visually distinct from non-header columns.

If HCOLID is not defined, there are no header columns.

If an application attempts to modify HCOLID to identify a column containing one or more choices that are currently selected, those choices are deselected.

**Domain** citem

**Access** Create, Modify, Reference

**Default** None.

* + 1. HROWID elattribute

**Definition** In a table gadget, any row whose ritem does not follow the value of HROWID (in the collating sequence of ^$WINDOW) is a header row. Header rows do not scroll and are displayed in a manner visually distinct from non-header rows.

If HROWID is not defined, there are no header rows.

If an application attempts to modify HROWID to identify a row containing one or more choices that are currently selected, those choices are deselected.

**Domain** ritem

**Access** Create, Modify, Reference

**Default** None.

* + 1. ID elattribute

**Definition** ID is an internal identifier for the element that assists in accessing this element in the underlying windowing platform.

**Domain** Unspecified

**Access** Reference

**Default** Not applicable

* + 1. INSELECT elattribute

**Definition** INSELECT indicates the insertion point within a text value and a count of the number of characters selected. New characters inserted into the text value are added immediately following the position indicated by the insert value.

If seloff is 0 (zero), no characters are selected. If seloff is greater than 0 (zero), the seloff characters immediately following the position identified by insert (that is, characters insert+1 through insert+seloff) are selected. If seloff is less than 0 (zero), the seloff characters immediately preceding the position identified by insert (that is, characters insert+1+seloff through insert) are selected.

When an application attempts to assign a value to the INSELECT elattribute, the following series of actions are performed. If the value of insert is less than 0 (zero), it is set to 0 (zero). If the value of insert exceeds the length of the text value, insert is set to indicate the last character of the text value. If the value of seloff would select characters before the beginning of or after the end of the data value, seloff is set, respectively, to indicate that the selection extends to the first or final character in the value.

If an application assigns values to a gadget's INSELECT and VALUE elattributes with a single M MERGE command, the result is as if VALUE were defined first.

**Domain** insel

**Access** Create, Modify, Reference

**Default** Platform

* + 1. INTERVAL elattribute

**Definition** INTERVAL specifies the time, in seconds, that a timer runs following initiation before generating a timer event.

If an application attempts to assign a value to INTERVAL which is not greater than 0 (zero), an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** numexpr

**Access** Create, Modify, Reference

**Default** None

* + 1. LEFTSHOW elattribute

**Definition** LEFTSHOW specifies which non-header column is displayed left most in the scrolling region of a table gadget. If the user scrolls the columns, the value of LEFTSHOW changes.

If the application assigns a value to LEFTSHOW, the non-header columns may scroll.

If the value assigned to LEFTSHOW exactly matches a citem subscript specified for a non-header column, the column identified by that subscript is displayed left most in the scrolling region.

If the application assigns a value to LEFTSHOW which does not exactly match a citem subscript specified for a non-header column, the first non-header column identified by a citem subscript that follows (in the collation sequence of ^$WINDOW) the specified LEFTSHOW value is displayed left most in the scrolling region. LEFTSHOW is then set to the subscript value of that non-header column.

If the application assigns a value to LEFTSHOW which does not exactly match a citem subscript specified for a non-header column, and which is not followed (in the collation sequence of ^$WINDOW) by any citem subscript specified for the gadget, the columns scroll (if necessary) so that the last column in the table is displayed in the visible area. LEFTSHOW is then set to the citem displayed left most in the scrolling region.

If the aggregate width of the columns is at least great enough to fill the scrolling region and the value assigned to LEFTSHOW would cause the scrolling region to be only partially filled, the underlying windowing platform or the MWAPI implementation may scroll the list to fill the scrolling region. LEFTSHOW is then set to the citem value for the column displayed left most in the scrolling region.

**Domain** citem or emptystring

**Access** Create, Modify, Reference

**Default** citem subscript of first non-header column in the table, if any; emptystring otherwise.

* + 1. LINKACTION elattribute

**Definition** Assigning a value to LINKACTION causes an action to occur in the element's Data Exchange link, if any. The assignment has no effect if the element does not have a Data Exchange link established. Actions occur asynchronously, and cause a LINKACTION event upon completion or timeout.

|  |  |
| --- | --- |
| POKE | The application at the other end of this element's Data Exchange link is asked to assign the text in the element's VALUE elattribute to the current item |
|  |  |
| REQUEST | The application at the other end of this element's Data Exchange link is asked to send the text of the current item. The text received is assigned to the element's VALUE elattribute. |
|  |  |
| EXECUTE | The application at the other end of this element's Data Exchange link is asked to execute the command in the element's VALUE elattribute. |
|  |  |
| emptystring | No action is required. |

Reference to this elattribute always returns emptystring.

**Domain** deaction

**Access** Create, Modify, Reference

**Default** Not applicable

* + 1. LINKITEM elattribute

**Definition** Specifies the item of interest in the Data Exchange source identified in the element's LINKSOURCE elattribute. For a TEXT gadget, data supplied by this item will be truncated at the first linend, if any.

**Domain** deitem

**Access** Create, Modify, Reference

**Default** emptystring

* + 1. LINKMODE elattribute

**Definition** The value of LINKMODE controls the mode of operation of the element's Data Exchange link, as defined by the elattributes LINKSOURCE and LINKITEM.

|  |  |
| --- | --- |
| NONE | No link is open |
|  |  |
| AUTOMATIC | The element's VALUE elattribute is updated whenever the item identified by LINKSOURCE and LINKITEM changes. |
|  |  |
| MANUAL | The element is not informed of changes in the value of the item identified by LINKSOURCE and LINKITEM |
|  |  |
| NOTIFY | The element is informed whenever the item identified by LINKSOURCE and LINKITEM changes, but the element's VALUE elattribute is not updated. |

**Domain** demode

**Access** Create, Modify, Reference

**Default** emptystring

* + 1. LINKSOURCE elattribute

**Definition** Specifies the data source of the element's Data Exchange link.

**Domain** desource

**Access** Create, Modify, Reference

**Default** emptystring

* + 1. LINKTIMEOUT elattribute

**Definition** LINKTIMEOUT specifies the time, in seconds, within which a data source is expected to respond. A LINKACTION event with a LINKSTATUS einfoattribute of “TIMEOUT” is generated if the data source does not respond within this interval.

If an M process attempts to assign a value to INTERVAL which is not greater than 0 (zero) an error condition occurs with $ECODE containing ",M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** numexpr

**Access** Create, Modify, Reference

**Default** Platform

* + 1. NEXTG elattribute

**Definition** NEXTG indicates the action that occurs if, while this gadget has focus, the user presses the key sequence that requests movement to the next gadget. The key sequence that requests movement to the next gadget is determined by the underlying windowing platform.

If an application attempts to assign a value to NEXTG which does not identify a gadget that has been defined for the same window, an error condition occurs with an ecode = "M49" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If the NEXTG elattribute for gadget G1 identifies a gadget G that cannot get focus, that gadget's NEXTG elattribute determines where focus goes. This process is repeated for successive Gs until (a) gadget G's NEXTG elattribute identifies gadget G1 or a previous gadget G, or gadget G's NEXTG elattribute is not specified, in which case focus remains at gadget G1; or (b) gadget G's NEXTG elattribute identifies a gadget Gn which can get focus is reached, in which case focus goes to gadget Gn.

The effect is not specified if NEXTG identifies a gadget that has been destroyed after the value identifying it was assigned to NEXTG.

**Domain** gname

**Access** Create, Modify, Reference

**Default** None

* + 1. POS elattribute

**Definition** POS specifies the position of the origin of the element with respect to the origin of its parent's application area, expressed in the unit of measure defined by the unitspec component of the pos value, if present, and by the element's UNITS elattribute otherwise. For a document, list box, list button, list entry box, long list box, radio button set or text gadget, the position excludes the gadget's title, if any. For a group frame, the position may exclude the gadget's title, if any.

If the unit of measure is CHAR, the basis font is the font specified by the parent's FFACE, FSIZE and FSTYLE wattributes. If the unit of measure is REL, the basis size is the size of the parent's window viewport when the parent is created.

For gadgets, if a value is not assigned to the POS elattribute when a gadget is created, or if a value is assigned which does not contain both hpos and vpos components, an error condition occurs with an ecode = "M52" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

For menus not displayed as menu bars or as submenus, if a value has not been assigned to the POS elattribute before the menu is made visible, or if a value is assigned which does not contain both hpos and vpos components, an error condition occurs with an ecode = "M52" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

For menus displayed as menu bars and for menus displayed as submenus, the value of the POS elattribute does not determine the position of the menu.

**Domain** pos

**Access** Create, Modify, Reference

**Default** Not applicable

* + 1. PTYPE elattribute

**Definition** PTYPE specifies the pointer appearance when the pointer is in the gadget. A gadget's PTYPE elattribute takes precedence over its parent window's PTYPE wattribute.

If the display on which the gadget appears does not have a pointer, the value of PTYPE is ignored.

**Domain** ptype

**Access** Create, Modify, Reference

**Default** None

* + 1. RESOURCE elattribute

**Definition** RESOURCE specifies an image to display in the gadget.

If, when a symbol gadget is created, an application does not assign a value to the RESOURCE elattribute, an error condition occurs with an ecode = "M52" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

If an application attempts to assign a value to the RESOURCE elattribute which is not valid, an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

The effect is not specified if the image identified by RESOURCE becomes invalid or unavailable after the value identifying it has been assigned to the RESOURCE elattribute.

**Domain** For a symbol gadget, symbol. For other gadgets, extresource.

**Access** Create, Modify, Reference

**Default** None

* + 1. ROW elattribute

**Definition** ROW specifies the rows in a table gadget, via descendant nodes of the form

^$W[INDOW]( wname , gkeyword , gname , rowkeyword , ritem )

where

rowkeyword ::= expr V ROW

No value is present at the level of the ROW elattribute itself. There is a value of emptystring at the ritem level.

A ROW elattribute descendant is present for every ritem subscript that is present as a descendant of the CHOICE elattribute for the gadget. If an application assigns a CHOICE descendant with an ritem subscript that is not present as a ROW descendant, a ROW descendant with that ritem subscript is automatically created.

A ROW elattribute descendant may be present with a ritem subscript which is not present as a descendant of the CHOICE elattribute for the gadget. Such a ritem subscript defines an empty row.

**Domain** emptystring

**Access** Create, Modify, Reference

**Default** Not Applicable.

* + 1. ROWCOL elattribute

**Definition** ROWCOL specifies how choices are arranged within rows and columns in a radio button set gadget. If a gadget contains N choices, they are displayed in a matrix of R rows by C columns. If rowcol is equal to "ROW", R is equal to rowcolsize and C is equal to the M numeric interpretation of

N\R+(N#R>0)

If rowcol is equal to "COL", C is equal to rowcolsize and R is equal to the M numeric interpretation of

N\C+(N#C>0)

If filldir is equal to "H", then rows are filled first. That is, all cells in row n are filled before any cells in row n+1 are filled. If filldir is equal to "V", then columns are filled first. That is, all cells in column n are filled before any cells in column n+1 are filled. It is possible to have empty cells, rows and columns. Rows and columns are filled beginning with the cell closest to the origin and proceeding in a positive direction.

If an application attempts to assign a rowcolsize value that is less than 1 (one), an error occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

All columns have the same width. If the hsize component of the SIZE elattribute is specified for the gadget, the column width is approximately equal to hsize/C. If the hsize component of the SIZE elattribute is not specified for the gadget, the column width is approximately equal to the widest choice specified for the gadget.

**Domain** rowcolspec

**Access** Create, Reference

**Default** If ROWCOL is not specified it defaults to "COL,1,V".

If intlit is not specified, it defaults to 1 (one).

If filldir is not specified, it defaults to "V".

* + 1. ROWHEIGHT elattribute

**Definition** ROWHEIGHT specifies the default height for rows in a gadget.

If the unit of measure is CHAR, the basis font is the font specified by the gadget's FFACE, FSIZE and FSTYLE elattributes. If the unit of measure is REL, the basis size is the size of the gadget at creation.

**Domain** height

**Access** Create, Modify, Reference

**Default** Platform.

* + 1. SCROLL elattribute

**Definition** SCROLL specifies scrolling behavior for a gadget's data.

If the value of the hsize component of SCROLL is greater than 0 (zero), horizontal scrolling can occur to reveal horizontal coordinates from 0 to hsize. If the value of the hsize component of SCROLL is equal to 0 (zero), horizontal scrolling can occur but no explicit horizontal range is specified for the data. If hsize is omitted, horizontal scrolling cannot occur.

If the value of the vsize component of SCROLL is greater than 0 (zero), vertical scrolling can occur to reveal vertical coordinates from 0 to vsize. If the value of the vsize component of SCROLL is equal to 0 (zero), vertical scrolling can occur but no explicit vertical range is specified for the data. If vsize is omitted, vertical scrolling cannot occur.

The unit of measure is determined in the same manner as is specified for the SIZE elattribute.

For a document gadget, the hsize component of SCROLL also specifies the width at which wrapping occurs.

A scroll bar may or may not be present when it is not required.

If an application attempts to assign a SCROLL value that contains a negative value for hsize or vsize, an error occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** size

**Access** Create, Reference

**Default** emptystring

* + 1. SCROLLBY elattribute

**Definition** SCROLLBY specifies the amount by which the VALUE elattribute increases or decreases when the user presses a scroll gadget's movement control. If the user presses a small movement control, the scroll gadget's VALUE elattribute is increased or decreased by the amount specified by the scrollsmall component of the SCROLLBY value. If the user presses a large movement control, the scroll gadget's VALUE elattribute is increased or decreased by the amount specified by the scrollbig component of the SCROLLBY value.

If an application attempts to assign a scrollby value that contains a negative value for scrollsmall or scrollbig, an error occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** scrollby

**Access** Create, Modify, Reference

**Default** For scrollsmall, the default is 1 (one).

For scrollbig, the default is equal to the M numeric interpretation of

X-Y/5

Where X and Y are equal to, respectively, the scrollmax and scrollmin components of the value of the gadget's SCROLLRANGE elattribute.

* + 1. SCROLLDIR elattribute

**Definition** SCROLLDIR specifies the orientation of a scroll gadget. If the value of SCROLLDIR is "H", the scroll gadget is displayed horizontally. If the value of SCROLLDIR is "V", the scroll gadget is displayed vertically.

**Domain** orientation

**Access** Create, Reference

**Default** "H"

* + 1. SCROLLPOS elattribute

**Definition** SCROLLPOS specifies the position of the position indicator in a long list box, relative to the scale established by the SCROLLRANGE elattribute. If SCROLLPOS is not defined, no position indicator is present in the long list box.

**Domain** numexpr

**Access** Create, Modify, Reference

**Default** None

* + 1. SCROLLRANGE elattribute

**Definition** SCROLLRANGE specifies the value range that a scroll gadget or long list box displays. The position of a scroll gadget's value indicator or of a long list box's position indicator is determined by the scroll value compared to the range specified by the SCROLLRANGE elattribute. The scrollmax component of the SCROLLRANGE value represents the position on the gadget that is farthest from the origin. For a scroll gadget, the scroll value is equal to the numeric value of the VALUE elattribute. For a long list box gadget, the scroll value is equal to the numeric value of the SCROLLPOS elattribute.

If an application attempts to assign a value to a scroll gadget's VALUE elattribute or to a long list box's SCROLLPOS elattribute that is greater than scrollmax, the VALUE or SCROLLPOS attribute is set to scrollmax. If an application attempts to assign a value to a scroll gadget's VALUE elattribute or to a long list box's SCROLLPOS elattribute that is less than scrollmin, the VALUE or SCROLLPOS attribute is set to scrollmin. If an application attempts to assign to a gadget's SCROLLRANGE elattribute a value with scrollmax less than or equal to scrollmin, the effect is not specified.

**Domain** scrollrange

**Access** Create, Modify, Reference

**Default** The default for scrollmax is 100. The default for scrollmin is 0 (zero).

* + 1. SELECTMAX elattribute

**Definition** For certain gadgets with choices, SELECTMAX specifies the maximum number of choices that can be concurrently selected. The user is prevented from selecting more than the specified number of choices. A value of 0 (zero) indicates that no limit is imposed.

**Domain** intexpr

**Access** Create, Reference

**Default** 1 (one)

* + 1. SELECTVAL elattribute

**Definition** SELECTVAL contains the selected data associated with or represented by the gadget.

If the gadget's INSELECT elattribute is not defined, its SELECTVAL elattribute is also not defined and is not modifiable. If the gadget's INSELECT elattribute is defined, its SELECTVAL elattribute is also defined. If the seloff component of the gadget's INSELECT elattribute is equal to zero, its SELECTVAL elattribute is emptystring.

If an application assigns a value, X, to SELECTVAL, the value of X replaces the selected data, if any, in the gadget's data value. If an application removes the SELECTVAL with the M KILL command, the selected data, if any, is removed from the gadget's data value. Any changes made to a gadget's data value via SELECTVAL are reflected in the gadget's VALUE elattribute and may also cause modification of the gadget's INSELECT elattribute.

**Domain** For a document gadget, longchars.

For a list entry box or text gadget, chars.

**Access** Modify, Reference

**Default** Not Applicable

* + 1. SIZE elattribute

**Definition** SIZE specifies the size of the gadget, expressed in the unit of measure defined by the unitspec component of the size value, if present, and by the gadget's UNITS elattribute otherwise.

If the unit of measure is not CHAR and a gadget is rectangular, SIZE specifies the overall size of the gadget. If the unit of measure is not CHAR and a gadget is not rectangular, SIZE specifies the size of the smallest rectangle that can bound the gadget.

If the unit of measure is CHAR, the basis font is the font specified by the gadget's FFACE, FSIZE and FSTYLE elattributes. If the FFACE elattribute is not applicable to the gadget, the basis font is the font specified by the gadget's TFFACE, TFSIZE and TFSTYLE elattributes. If the TFFACE elattribute is not applicable to the gadget, the basis font is the font specified by the parent window's FFACE, FSIZE and FSTYLE wattributes. For a check box, document, label, list box, list button, list entry box, long list box, push button, radio button set, or text gadget, if the unit of measure is CHAR, the gadget is large enough to accommodate vsize lines of hsize characters each using the basis font; the overall size of the gadget may be larger. For other gadget types, a unit of measure of CHAR specifies the overall size of the gadget.

If the unit of measure is REL, the basis size is the size of the parent's window viewport when the parent is created.

For a group frame, hsize and vsize values greater than 0 (zero) produce a rectangle; an hsize value of 0 (zero) produces a vertical line; a vsize value of 0 (zero) produces a horizontal line. If an application attempts to assign to the SIZE elattribute of a group frame a size value for which hsize and vsize are both 0 (zero), an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

For a generic box or group frame gadget, if a value is not assigned to the SIZE elattribute when the gadget is created, an error condition occurs with an ecode = "M52" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

For a generic box or group frame gadget, if an application attempts to assign to the SIZE elattribute a value which does not contain both an hsize component and a vsize component, an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

Whenever a size value specifies the overall size of a gadget, the size includes the gadget's border, if any. For a check box, label or push button gadget, the overall size of a gadget includes the gadget's title, if any. For a document, list box, list button, list entry box, long list box, radio button set or text gadget, the overall size of a gadget excludes the gadget's title, if any. For a group frame, the overall size of a gadget may exclude the gadget's title, if any.

For a list button, size includes the size of the button and the list. The actual height of the list button gadget may be whatever is required to contain the choices and some platforms may not allow the list to be scrolled.

For a symbol gadget, if the SIZE elattribute value assigned by an application is smaller than the size necessary to display the image specified for the gadget, the image is cropped. If the SIZE elattribute value assigned by an application is larger than the size necessary to display the image specified for the gadget, the image may be stretched to fit the assigned size or the gadget's size may be reduced to the actual size necessary to display the image.

Assigning a value to the SIZE elattribute for a gadget does not imply any automatic scaling of the gadget's contents.

When an application assigns a value to the SIZE elattribute of an existing gadget, if hsize or vsize is not specified then the horizontal or vertical size of the gadget, respectively, is not altered.

**Domain** size

**Access** Create, Modify, Reference

**Default** For a check box, document, symbol, label, list box, list button, list entry box, long list box, push button, radio button set, scroll , or text element, if hsize, vsize, or both are not specified when a gadget is created, the default for each unspecified value is platform. For a symbol, if hsize, vsize, or both are not specified when a gadget is created, the default for each unspecified value is the size necessary to display the image specified for the gadget.

* + 1. TBCOLOR elattribute

**Definition** TBCOLOR specifies the background color for the gadget's title. If the underlying windowing platform does not enable the title to have a different background color from the remainder of the gadget, the value of the TBCOLOR elattribute is ignored.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent window's TBCOLOR wattribute, if any. Otherwise, value of parent windows's COLOR wattribute

* + 1. TFCOLOR elattribute

**Definition** TFCOLOR specifies the color for the gadget's title text. If the underlying windowing platform does not enable the title text to have a different color from the remainder of the gadget's text, the value of the TFCOLOR elattribute is ignored.

**Domain** color

**Access** Create, Modify, Reference

**Default** Value of parent window's TFCOLOR wattribute, if any. Otherwise, value of gadget's FCOLOR elattribute

* + 1. TFFACE elattribute

**Definition** TFFACE specifies the type face used for the gadget's title text.

**Domain** fontface

**Access** Create, Modify, Reference

**Default** Value of parent window's FFACE wattribute

* + 1. TFSIZE elattribute

**Definition** TFSIZE specifies the type size used for the gadget's title text.

**Domain** fontsize

**Access** Create, Modify, Reference

**Default** Value of parent window's FSIZE wattribute

* + 1. TFSTYLE elattribute

**Definition** TFSTYLE specifies the type style used for the gadget's title text.

**Domain** fontstyle

**Access** Create, Modify, Reference

**Default** Value of parent window's FSTYLE wattribute

* + 1. TITLE elattribute

**Definition** TITLE specifies text that appears within or adjacent to a gadget.

If the value of the TITLE elattribute contains a mnemonichar and the gadget is of a type that can get focus, the character following the mnemonichar is a mnemonic character that may enable the user, via the keyboard, to give focus to the gadget and the mnemonichar is not displayed. If two mnemonichars are immediately adjacent within a TITLE value, no mnemonic character is defined and a single mnemonichar appears in the displayed title text. The character mnemonichar cannot itself be used as a mnemonic character.

If the value of the TITLE elattribute contains a mnemonichar and the gadget is not of a type that can get focus, then if the gadget's NEXTG elattribute is defined, focus is set according to the NEXTG value. If the gadget's NEXTG elattribute is not defined, use of the mnemonichar does not cause focus to change.

If the TITLE elattribute of a radio button set contains a mnemonichar, the behavior resulting from use of the mnemonic character is platform specific.

In addition to giving focus to a gadget, use of a mnemonic character will cause any additional platform-specific behavior to occur.

For a check box gadget, TITLE specifies the text that appears adjacent to the selection area.

For a group frame gadget, TITLE specifies text that appears within or overlaying the frame.

For a label gadget, TITLE specifies the label text.

For a document, list box, list button, list entry box, long list box, radio button set, scroll, or text gadget, TITLE specifies text that appears adjacent to the gadget.

For a push button, TITLE specifies the text that appears on the button.

**Domain** chars

**Access** Create, Modify, Reference

**Default** For a label gadget, TITLE defaults to an emptystring. For other gadget types, TITLE defaults to none.

* + 1. TOPSHOW elattribute

**Definition** TOPSHOW specifies which choice is displayed at the top of the visible area of certain gadget types that display choices, or which non-header row is displayed at the top of the scrolling region of a table gadget. If the user scrolls the choices or table rows, the value of TOPSHOW changes.

If the application assigns a value to TOPSHOW, the list of choices or table rows may scroll.

If the value assigned to TOPSHOW exactly matches an item subscript specified for a choice or an ritem subscript specified for a table non-header row, the choice or row identified by that subscript is displayed at the top of the visible area or scrolling region.

If the application assigns a value to TOPSHOW which does not exactly match an item subscript or an ritem subscript for a non-header row specified for the gadget, the choice row identified by the first item or the first non-header row identified by an ritem subscript that follows (in the collation sequence of ^$WINDOW) the specified TOPSHOW value is displayed at the top of the visible area or scrolling region. TOPSHOW is then set to that subscript value.

If the application assigns a value to TOPSHOW which does not exactly match an item or ritem subscript specified for the gadget, and which is not followed (in the collation sequence of ^$WINDOW) by any item or by any ritem subscript for a non-header row specified for the gadget, the list of choices or rows scrolls (if necessary) so that the last choice in the list or the last row in the table is displayed in the visible area. TOPSHOW is then set to the item or ritem displayed at the top of the visible area.

If the aggregate height of choices or rows is at least great enough to fill the display area and the value assigned to TOPSHOW would cause the display area to be only partially filled, the underlying windowing platform or the MWAPI implementation may scroll the list to fill the visible area. TOPSHOW is then set to the item or ritem value for the choice or non-header row displayed at the top of the visible area.

**Domain** For a table gadget

ritem or emptystring

For other gadget types

item or emptystring

**Access** Create, Modify, Reference

**Default** For a table gadget

ritem subscript of first row in the table, if any; emptystring otherwise.

For other gadget types

item subscript of first choice in the list, if any; emptystring otherwise.

* + 1. TPOS elattribute

**Definition** TPOS specifies the position of a title relative to the gadget it is associated with.

**Domain** titlepos

**Access** Create, Modify, Reference

**Default** "TOP"

* + 1. TYPE elattribute

**Definition** TYPE specifies the type of the gadget. If an application does not assign a value for the TYPE elattribute when attempting to create a gadget, an error condition occurs with an ecode = "M52" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** gtype

**Access** Create, Reference

**Default** Not applicable

* + 1. UNITS elattribute

**Definition** UNITS specifies the unit of measure for the element.

**Domain** unitspec

**Access** Create, Modify, Reference

**Default** Value of parent's UNITS wattribute

* + 1. VALUE elattribute

**Definition** VALUE specifies the data associated with or represented by the gadget.

For a check box gadget, if the user selects the gadget, VALUE is set to 1 (one). If the user deselects the gadget, VALUE is set to 0 (zero). If a value that is assigned by the application evaluates to TRUE, the gadget is selected. If a value that is assigned by the application evaluates to FALSE, the gadget is deselected.

For a document, list entry box or text gadget, VALUE specifies the text value. Within a document text value, each linend indicates that, when the text value is displayed, the character following the linend should begin a new line. Otherwise, the formatting mechanisms used for displaying a document text value are determined by the underlying windowing platform, if there is one, and by the MWAPI otherwise.

For a list box, long list box or table gadget, VALUE specifies the number of choices selected. For a list box or, long list box, the choices selected are indicated by descendant nodes, of the form

^$W[INDOW] ( wname , gkeyword , gname , vkeyword , item )

where

vkeyword ::= expr V VALUE

and item identifies a selected choice.

For a table gadget, the choices selected are indicated by descendant nodes, of the form

^$W[INDOW] ( wname , gkeyword , gname , vkeyword , ritem ,

citem )

where ritem and citem identify a selected choice.

For a list box, long list box or table gadget, the value of these nodes is the emptystring. If an application attempts to create a descendant node of VALUE which has an item that does not identify a defined choice for the gadget, an error condition occurs with an ecode = "M48" and $EREF containing a value that indicates the ssvn reference for which the error occurred. If an application attempts to create a descendant node of VALUE which has a citem or ritem that does not identify a non-header row or column for the gadget, an error condition occurs with an ecode = "M48" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

For a list box, long list box, or table gadget, an application can assign values to descendant nodes of VALUE, but it cannot assign a value to the VALUE node itself. However, an application can use an M KILL command of the form

^$W[INDOW] ( wname , gkeyword , gname , vkeyword )

to deselect all current selections in a list box, long list box, or table.

For a radio button set or list button gadget, VALUE specifies the item value of the selected choice. If an application attempts to assign to VALUE an item that does not identify a defined choice for the gadget, an error condition occurs with an ecode = "M48" and $EREF containing a value that indicates the ssvn reference for which the error occurred. If no choice is selected, the value of VALUE is an emptystring.

For a scroll gadget, VALUE specifies the value represented by the value indicator.

**Domain** For a check box gadget, truefalse.

For a document gadget, longchars.

For a list entry box or text gadget, chars.

For a list box, long list box, or table gadget, VALUE has a domain of intexpr and descendant nodes have a domain of emptystring.

For a radio button set or list button gadget, item or emptystring.

For a scroll gadget, numexpr.

**Access** Create, Modify, Reference

**Default** For a check box gadget, 0 (zero).

For a document, list button, list entry box, radio button set, or text gadget, emptystring.

For a list box , long list box, or table gadget, 0 (zero) with no descendant nodes.

For a scroll gadget, the value of the scrollmin component of the SCROLLRANGE elattribute.

* + 1. VISIBLE elattribute

**Definition** If VISIBLE is FALSE, a gadget or menu cannot be seen by the user. If VISIBLE is TRUE, a gadget or menu can be seen by the user provided that its ancestors' VISIBLE attributes are TRUE, it is not obscured by another window, and it is not positioned outside the visible area of its parent.

For menus displayed as menu bars and for menus displayed as submenus, the value of the VISIBLE elattribute is ignored. A menu displayed as a menu bar is visible when its parent window is visible. A menu, S, displayed as a submenu becomes visible automatically when a menu choice whose SUBMENU cattribute identifies menu S is activated. A menu, S, displayed as a submenu becomes invisible when (a) a select event occurs for a menu choice of S for which the SUBMENU gattribute is not specified or (b) a menu displayed as a submenu of S becomes invisible.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* 1. Choice Attributes

Choice attributes, identified by cattributes, define the characteristics of choices. For table gadgets, choice specifications have the form

^$W[INDOW] ( wname , gkeyword , gname , ckeyword , ritem , citem , cattribute )

ckeyword ::= expr V CHOICE

For other gadgets, choice specifications have the form

^$W[INDOW] ( wname , gkeyword , gname , ckeyword , item , cattribute )

For menus, choice specifications have the form

^$W[INDOW] ( wname , mkeyword , mname , ckeyword , item , cattribute )

|  |  |
| --- | --- |
| cattribute ::= expr V | ACCELERATOR  ACTIVE  AID  EVENT  MARKER  RESOURCE  SEPARATOR  SUBMENU  applicationattribute  implementationattribute |

* + 1. Applicability of Choice Attributes

The applicability of cattributes to element types is specified by Table 3. A "Yes" entry indicates that the use of the attribute with the element type is defined. An empty entry indicates that the use of the attribute with the element type is reserved. implementationattribute and applicationattribute values are applicable to all element types.

| **Table 3: Choice Attributes****Choice Attributes** | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | List  Box | List  Button | List  Entry  Box | Long  List  Box | Menu | Radio  Button  Set | Table |
| ACCELERATOR |  |  |  |  | Yes |  |  |
| ACTIVE | Yes | Yes | Yes | Yes | Yes | Yes |  |
| AID | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| EVENT |  |  |  |  | Yes |  |  |
| MARKER |  |  |  |  | Yes |  |  |
| RESOURCE |  |  |  |  |  |  | Yes |
| SEPARATOR |  |  |  |  | Yes |  |  |
| SUBMENU |  |  |  |  | Yes |  |  |

* + 1. ACCELERATOR cattribute

**Definition** ACCELERATOR specifies an accelerator key sequence for a choice in a menu. Accelerator keys provide a means of selecting menu items via a keyboard.

**Domain** keycode

**Access** Create, Reference

**Default** None

* + 1. ACTIVE cattribute

**Definition** If ACTIVE is FALSE, the choice is disabled. The user cannot select or deselect the choice, and it may be displayed grayed or in some other fashion to indicate that it is disabled.

If ACTIVE is TRUE, the choice is enabled.

If a value equal to FALSE is assigned to the ACTIVE cattribute for a choice which is selected, the choice is deselected. If an application attempts to select a choice which has an ACTIVE cattribute value equal to FALSE, an error condition occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* + 1. AID cattribute

**Definition** The AID cattribute is not used by the MWAPI. It is provided so that an application can associate an application-dependent value with a menu, list, radio button, or table choice.

**Domain** chars

**Access** Create, Modify, Reference

**Default** None

* + 1. EVENT cattribute

**Definition** EVENT specifies, via descendant nodes identified by etype subscripts, the choice's callback routine(s) for the event specified by etype. The format of a choice event specification is defined in Section 9.2. No value is present at the level of the EVENT cattribute itself.

**Domain** For immediate descendants

L doargument

**Access** For immediate descendants

Create, Modify, Reference

**Default** None

* + 1. MARKER cattribute

**Definition** MARKER specifies a marker image that appears to the left of the choice text for a menu item. In the case of a menu-bar menu item, support for such a marker is platform-dependent.

**Domain** marker

**Access** Create, Modify, Reference

**Default** None

* + 1. RESOURCE cattribute

**Definition** RESOURCE specifies an image to be displayed in a cell.

**Domain** extresource

**Access** Create, Modify, Reference

**Default** None

* + 1. SEPARATOR cattribute

**Definition** In a pop-up menu, the SEPARATOR cattribute causes a visual separator (typically a single horizontal line) to appear below the choice text of the choice for which it is specified.

In a menu bar, the first instance of the SEPARATOR cattribute causes subsequent menu choices to appear right justified within the menu bar, if appropriate for the underlying windowing platform. Any instances of the SEPARATOR cattribute after the first within a menu bar are ignored.

The only value that can be assigned to the SEPARATOR cattribute is an emptystring.

**Domain** emptystring

**Access** Create, Reference

**Default** None

* + 1. SUBMENU cattribute

**Definition** SUBMENU identifies the mname of the pop-up menu to display if a menu choice is selected. If the mname assigned to the SUBMENU cattribute does not identify a menu defined for the same window or if the mname assigned to the SUBMENU is equal to the mname of the menu itself, an error condition occurs with an ecode = "M48" and $EREF containing a value that indicates the ssvn reference for which the error occurred. If the SUBMENU cattribute is not specified for a menu choice, no menu is displayed by the MWAPI if the user selects the menu choice.

**Domain** mname

**Access** Create, Modify, Reference

**Default** None

* 1. Row and Column Attributes

Row attributes, identified by rowattributes, define the characteristics of rows for certain gadgets. Row specifications have the form

^$W[INDOW] ( wname , gkeyword , gname , rowkeyword , ritem , rowattribute )

rowkeyword ::= expr V ROW

|  |  |
| --- | --- |
| rowattribute ::= expr V | ROWHEIGHT  implementationattribute |

Column attributes, identified by colattributes, define the characteristics of columns for certain gadgets. Column specifications have the form

^$W[INDOW] ( wname , gkeyword , gname , colkeyword , citem , colattribute )

colkeyword ::= expr V COL

|  |  |
| --- | --- |
| colattribute ::= expr V | COLWIDTH  implementationattribute |

* + 1. Applicability of Row and Column Attributes

rowattributes and colattributes are applicable to table gadgets. Their use with all other element types is reserved.

* + 1. COLWIDTH colattribute

**Definition** COLWIDTH specifies the width of a column.

If a width value assigned to COLWIDTH does not include a unitspec component, the unit of measure is determined by the gadget's UNITS elattribute. If the unit of measure is CHAR, the basis font is the font specified by the gadget's FFACE, FSIZE and FSTYLE elattributes. If the unit of measure is REL, the basis size is the size of the gadget at creation.

**Domain** width

**Access** Create, Modify, Reference

**Default** Value of gadget's COLWIDTH elattribute.

* + 1. ROWHEIGHT rowattribute

**Definition** ROWHEIGHT specifies the height of a row.

If a height value assigned to ROWHEIGHT does not include a unitspec component, the unit of measure is determined by the gadget's UNITS elattribute. If the unit of measure is CHAR, the basis font is the font specified by the gadget's FFACE, FSIZE and FSTYLE elattributes. If the unit of measure is REL, the basis size is the size of the gadget at creation.

**Domain** height

**Access** Create, Modify, Reference

**Default** Value of gadget's ROWHEIGHT elattribute.

# 

# Drawing

A drawing is specified by assigning drawcommand values to immediately descendant nodes of the DRAW attribute. These nodes have the form:

^$W[INDOW] ( wname , gkeyword , gname , dkeyword , drawnum )

gname is the identifier for a gadget and wname is the identifier for the window that is the parent of that gadget.

dkeyword ::= expr V DRAW

drawnum ::= numexpr

* 1. Draw Command Numbers

drawcommands are stored as immediate descendants of the DRAW elattribute using drawnum subscripts. The drawing that appears is that which would result from sequentially performing all drawcommands for all drawnum subscripts, beginning with the lowest valued drawnum.

If an application attempts to assign a drawnum subscript which has a negative value, an error occurs with an ecode = "M47" and $EREF containing a value that indicates the ssvn reference for which the error occurred.

* 1. Draw Commands

drawcommand specifies the draw commands that are available for a gadget if the gadget's DRAWTYPE elattribute has a value equal to "MDRAW".

|  |  |
| --- | --- |
| drawcommand ::= | drawarc  drawbcolor  drawbitmap  drawbox  drawfcolor  drawfface  drawfillpat  drawfsize  drawfstyle  drawline  drawmode  drawoval  drawpensize  drawpicture  drawpie  drawpoint  drawtext  drawunits  implementationdrawcommand |

When used in a drawcommand, the unit of measure for the following position or size specifications

hpos

vpos

radius

hpos1

vpos1

hpos2

vpos2

is determined by the preceding drawunits drawcommand. Prior to the first drawunits drawcommand for the gadget, the unit of measure is determined by the value of the gadget's UNITS elattribute. The unit of measure determines how size and position values are converted to pixels.

Positions are relative to the gadget’s origin.

When used in a drawcommand, the unit of measure for the following angle specifications

startangle

endangle

is degrees with a 0 (zero) value representing a position on the positive horizontal axis. Degrees increase in a counterclockwise direction.

* + 1. ARC Draw Command

drawarc ::= expr V ARC , hpos , vpos , radius , startangle , endangle

radius ::= numlit

startangle ::= numlit

endangle ::= numlit

The drawarc drawcommand draws a curved line using the current draw foreground color, draw mode, and pen size. The curved line is the portion of the circumference of a circle, with its center at position hpos,vpos and a radius of radius, that lies between the angles startangle and endangle.

* + 1. BCOLOR Draw Command

drawbcolor ::= expr V BCOLOR , color , fillmode

The drawbcolor drawcommand specifies the current draw background color. If fillmode is equal to "OPAQUE", previous components of the drawing cannot be seen through the draw background color. If fillmode is equal to "TRANSPARENT", previous components of the drawing can be seen through the draw background color.

The specified color is used by subsequent drawcommands until another drawbcolor drawcommand is specified. Prior to the first drawbcolor drawcommand for a drawing, the current draw background color is determined by the element's BCOLOR elattribute and the current fill mode is TRANSPARENT.

* + 1. BITMAP Draw Command

drawbitmap ::= expr V BITMAP , hpos , vpos , extresource

The drawbitmap drawcommand causes the bitmap image identified by extresource to be displayed with its origin at position hpos,vpos.

* + 1. BOX Draw Command

drawbox ::= expr V BOX , hpos1 , vpos1 , hpos2 , vpos2

hpos1 ::= numlit

vpos1 ::= numlit

hpos2 ::= numlit

vpos2 ::= numlit

The drawbox drawcommand draws the perimeter of a rectangle using the current draw foreground color, draw mode, and pen size. One corner of the rectangle is at position hpos1,vpos1. The diagonally opposite corner of the rectangle is at position hpos2,vpos2. The interior of the rectangle is filled using the current draw background color and fill pattern.

* + 1. DMODE Draw Command

drawmode ::= expr V DMODE , dmode

The drawing mode determines the effect of certain drawcommands on pixels in the drawing area. If the value of dmode is equal to "SOLID", drawcommands cause pixels to be changed to the current draw foreground color or draw background color. If the value of dmode is equal to "XOR", the new color of each pixel is determined by performing a Boolean exclusive or (XOR) between the pixel in the drawing area and the current draw foreground color or draw background color.

The drawmode drawcommand specifies the current drawing mode. The specified mode is used by subsequent drawcommands until another drawmode drawcommand is specified. Prior to the first drawmode drawcommand for a drawing, the current drawing mode is SOLID.

* + 1. DRAWTEXT Draw Command

drawtext ::= expr V DRAWTEXT , hpos , vpos , chars

The drawtext drawcommand displays the characters in chars using the current font face, font size, font style, and draw foreground color. The background of the character cells is filled using the current draw background color if the current fill mode is OPAQUE. The position hpos,vpos specifies the location of the upper left corner of the character cell for the first character in chars.

* + 1. FCOLOR Draw Command

drawfcolor ::= expr V FCOLOR , color

The drawfcolor drawcommand specifies the current drawing foreground color. The specified color is used by subsequent drawcommands until another drawfcolor drawcommand is specified. Prior to the first drawfcolor drawcommand for a drawing, the current draw foreground color is determined by the gadget’s FCOLOR elattribute.

* + 1. FFACE Draw Command

drawfface ::= expr V FFACE , fontface

The drawfface drawcommand specifies the current font face. The specified font face is used by subsequent drawcommands until another drawfface drawcommand is specified. Prior to the first drawfface drawcommand for a drawing, the current font face is specified by the gadget's FFACE elattribute. Requesting of a new font from the underlying windowing platform (if any) is deferred until the first subsequent drawcommand requiring a font.

* + 1. FILLPAT Draw Command

drawfillpattern ::= expr V FILLPAT , fillpattern

The drawfillpattern drawcommand specifies the current fill pattern, as follows:

|  |  |
| --- | --- |
| **fillpattern Value** | **Pattern** |
| BDIAGONAL | Diagonal lines running from lower right to upper left. |
| CROSSHATCH | Vertical and horizontal lines. |
| DCROSSHATCH | Diagonal lines running from lower right to upper left and diagonal lines running from lower left to upper right. |
| FDIAGONAL | Diagonal lines running from lower left to upper right. |
| HORIZONTAL | Horizontal lines. |
| NONE | Filling is turned off. |
| SOLID | The area is filled with the current draw foreground color. |
| VERTICAL | Vertical lines. |

If the value of drawfillpattern is an implementationvalue, an implementation-specific fill pattern is used.

The fill pattern is drawn using the current draw foreground color and draw mode, and fills the area. Regions of the area not drawn using the current drawing foreground color are drawn using the current background drawing color and draw mode.

The specified pattern is used by subsequent drawcommands until another drawfillpattern drawcommand is specified. Prior to the first drawfillpattern drawcommand for a drawing, the current fill pattern is determined by the underlying windowing platform.

* + 1. FSIZE Draw Command

drawfsize ::= expr V FSIZE , fontsize

The drawfsize drawcommand specifies the current font size. The specified font size is used by subsequent drawcommands until another drawfsize drawcommand is specified. Prior to the first drawfsize drawcommand for a drawing, the current font size is specified by the gadget's FSIZE elattribute. Requesting of a new font from the underlying windowing platform (if any) is deferred until the first subsequent drawcommand requiring a font.

* + 1. FSTYLE Draw Command

drawfstyle ::= expr V FSTYLE , fontstyle

The drawfstyle drawcommand specifies the current font style. The specified font style is used by subsequent drawcommands until another drawfstyle drawcommand is specified. Prior to the first drawfstyle drawcommand for a drawing, the current font size is specified by the gadget's FSTYLE elattribute. Requesting of a new font from the underlying windowing platform (if any) is deferred until the first subsequent drawcommand requiring a font.

* + 1. LINE Draw Command

drawline ::= expr V LINE , hpos1 , vpos1 , hpos2 , vpos2

The drawline drawcommand draws a line from the position hpos1,vpos1 to, but not including, the position hpos2,vpos2 using the current draw foreground color, draw mode, and pen size.

* + 1. OVAL Draw Command

drawoval ::= expr V OVAL , hpos1 , vpos1 , hpos2 , vpos2

The drawoval drawcommand draws an ellipse using the current draw foreground color, draw mode, and pen size. The ellipse is drawn tangent to an invisible rectangle with one corner at hpos1,vpos1 and the diagonally opposite corner at hpos2,vpos2. The ellipse is filled using the current draw background color and fill pattern.

* + 1. PENSIZE Draw Command

drawpensize ::= expr V PENSIZE , pensize

pensize ::= intlit

The drawpensize drawcommand specifies the current pen size. pensize indicates the diameter, measured in pixels, of a round pen. If pensize is equal to 0 (zero), no apparent drawing is done with the pen.

For the drawarc, drawpoint, and drawline drawcommands, the pen is centered on the line or point. For drawbox, drawoval, and drawpie drawcommands, the outer edge of the pen is tangent to the inner edge of the perimeter of the figure.

The specified size is used by subsequent drawcommands until another drawpensize drawcommand is specified. Prior to the first drawpensize drawcommand for a drawing, the current pen size is determined by the underlying windowing platform.

* + 1. PICTURE Draw Command

drawpicture ::= expr V PICTURE , hpos , vpos , extresource

The drawpicture drawcommand causes the picture identified by extresource to be displayed with its origin at position hpos,vpos.

* + 1. PIE Draw Command

drawpie ::= expr V PIE , hpos , vpos , radius , startangle , endangle

The drawpie drawcommand draws a curved line and two straight lines using the current draw foreground color, draw mode, and pen size and fills the enclosed area using the current draw background color and fill pattern. The curved line is the portion of the circumference of a circle, with its center at position hpos,vpos and a radius of radius, that lies between the angles startangle and endangle. The straight lines connect the ends of the arc with the center of the circle.

* + 1. POINT Draw Command

drawpoint ::= expr V POINT , hpos , vpos

The drawpoint drawcommand draws a point at the location hpos,vpos using the current draw foreground color, draw mode, and pen size.

* + 1. UNITS Draw Command

drawunits ::= expr V UNITS , unitspec

The drawunits drawcommand specifies the unit of measure for subsequent drawcommands. If the unit of measure is CHAR, the basis font is the font specified by the most recent set of FFACE, FSIZE and FSTYLE drawcommands. If the unit of measure is REL, the basis size is the size of the gadget at creation.

# 

# Data Exchange

Using the Data Exchange capabilities of MWAPI an M process can access items in a data source and can itself provide data sources. In either case a *link* must be established between the user and provider of a data source. The user of a data source, or *client,* initiates a link, while the provider, or *server*, responds to such initiation.

## Server

To act as a server, the M process must create application windows that have the LINKTOPIC attribute defined, and which are on a display for which the LINKSERVICE attribute is defined. The value of these attributes yield a platform-dependent data source identifier for each such window. While the window exists, other applications can establish links with it, and get and set items within the data source. An M process may be involved in multiple links simultaneously.

The items in the data source offered by the M process are the VALUE attributes of all the gadgets that belong to the window and which have type DOCUMENT or TEXT. The items are identified by their gnames. Depending on the type of link established, the gadgets may respond to value changes by notifying their client or by sending their value to the client. The client can also ask a window to cause a LINKEXECUTE event, enabling an external application to execute M code, for example. See Section 9.1.9 for details of data exchange events.

A single client may use a single link to interact with more than one item on the same M data source. A single client may also establish more than one link to the same data source.

## Client

To act as a client, the M process uses the elattributes LINKACTION, LINKITEM, LINKMODE, LINKSOURCE and LINKTIMEOUT, to establish a link with a data source. The available data sources for a display are obtainable through the LINKSOURCE display attribute. LINKSOURCE identifies the data source, and LINKITEM identifies the item within that source. Setting LINKMODE establishes or breaks a link between the VALUE attribute of the gadget and the item in the data source. Depending on the value of LINKMODE, the gadget can be updated automatically as the data source item changes, or the LINKNOTIFY event can be caused. Any one gadget can only be associated with a single data source item, but an M process can have many links active simultaneously. LINKACTION allows the M program to drive the link, getting and setting values and executing commands in the server application as required. These operations are performed asynchronously and the outcome is notified using events. See Section 9.1.9 for details of data exchange events.

When an M process uses the appropriate elattributes on more than one gadget to connect as a client to the same data source, each gadget establishes a separate link.

An M process cannot establish a link with a data source provided by the same M process, nor do such data sources appear in the LINKSOURCE display attribute of that M process.

# Events

Events occur as a result of user actions or other happenings.

* 1. Event Types

The MWAPI defines the following event types, specified via etype values.

|  |  |
| --- | --- |
| etype ::= expr V | CHANGE  CHAR  CLICK  CLOSE  DBLCLICK  DESELECT  FOCUSGOBOTTOM  GODOWN  GODOWNBIG  GOTOP  GOUP  GOUPBIG  HELP  KEYDOWN  KEYUP LINKACTION LINKCLOSE LINKEXECUTE LINKOPEN LINKNOTIFY LINKPOKE LINKREQUEST LINKUPDATE  MAX  MIN  MOVE  PDOWN  PDRAG  PMOVE  PUP  RESIZE  RESTORE  SELECT  TIMER  UNFOCUS  implementationattribute |

The following event types, specified via betype values apply to build-mode windows.

|  |  |
| --- | --- |
| betype ::= expr V | CHAR  CLICK  CLOSE  DBLCLICK  FOCUS GDESELECT GMOVE  GRESIZE GSELECT  HELP  KEYDOWN  KEYUP  MAX MCOMMAND  MIN  MOVE  PDOWN  PDRAG  PMOVE  PUP  RESIZE  RESTORE  TRACK  UNFOCUS  implementationattribute |

* + 1. Window State Events

Window state events (i.e. close, minimize, restore, maximize, resize, and move events) occur when certain characteristics of a window change as a result of user action.

A close event, indicated by an etype value equal to "CLOSE", occurs for a window in response to a request to destroy the window.

A minimize event, indicated by an etype value equal to "MIN", occurs for a window when it changes from the open state to the iconic state.

A restore event, indicated by an etype value equal to "RESTORE", occurs for a window when it changes from the iconic state to the open state.

A maximize event, indicated by an etype value equal to "MAX", occurs for a window when it has been expanded to its maximum size.

A resize event, indicated by an etype value equal to "RESIZE", occurs for a window when its size changes. A resize event does not occur when a window is minimized, restored, or maximized.

A move event, indicated by an etype value equal to "MOVE", occurs for a window when its position changes. A move event also occurs for a window when it is resized if the window's origin changes, unless the window is in build mode.

* + 1. Pointer Events

Pointer events (i.e. click, double click, pointer down, pointer up, pointer move, and pointer drag events) occur when the user takes certain actions with the pointer and the pointer is located within the viewport of a window created via the MWAPI. If the pointer is within a gadget for which a pointer event can occur, then the event occurs for that gadget. Otherwise, if the pointer is within a window for which a pointer event can occur, then the pointer event occurs for that window.

A pointer down event, indicated by an etype value equal to "PDOWN", occurs when the user presses a pointer button.

A pointer up event, indicated by an etype value equal to "PUP", occurs when the user releases a pointer button. A pointer up event occurs for the window or gadget in which the pointer was located when the preceding pointer down event occurred, even if the pointer was subsequently dragged outside of that window or gadget.

A pointer move event, indicated by an etype value equal to "PMOVE", occurs when the user moves the pointer with no pointer buttons pressed.

For a scroll gadget, a pointer drag event, indicated by an etype value equal to "PDRAG", occurs when the user drags the gadget's position indicator. For windows and other gadget types, a pointer drag event occurs when the user moves the pointer with at least one pointer button pressed. Pointer drag events occur for the window or gadget in which the pointer was located when the drag was initiated, even if the pointer is subsequently dragged outside of the window or gadget in which the drag began.

A click event, indicated by an etype value equal to "CLICK", occurs when the user presses and releases a pointer button without an intervening change in the pointer position. For a button click action by the user, the following sequence of events occurs: pointer down, click, and pointer up.

A double click event, indicated by an etype value equal to "DBLCLICK", occurs when (a) the user twice presses and releases a pointer button without any intervening changes in the pointer position and (b) the second button press occurs within a specified time interval of the first button press. The time interval is not specified by the MWAPI and may be controllable by the user. For a button double click action by the user, the following sequence of events occurs: pointer down, click, pointer up, pointer down, double click, and pointer up.

A pointer event does not occur if the FILTERIN especattribute is present for the event and the pbuttoncode associated with the event is not one of the pbuttoncodes present in the value of that FILTERIN especattribute.

No behavior is defined for events related to pointer buttons which are not mapped to pointer button codes.

* + 1. Keyboard Events

Keyboard events (i.e. key down, key up, and character events) occur when the user presses or releases one or more keyboard keys.

If focus is on a gadget for which a keyboard event can occur, then the keyboard event occurs for that gadget. Otherwise, if focus is on a window for which a keyboard event can occur, then the keyboard event occurs for that window.

No behavior is defined for key sequences which are not mapped to key codes or character codes.

A key down event, indicated by an etype value equal to "KEYDOWN", occurs when the user presses a key.

A key up event, indicated by an etype value equal to "KEYUP", occurs when the user releases a key.

A character event, indicated by an etype value equal to "CHAR", occurs when one or more key transitions results in the production of a character.

A keyboard event does not occur if (a) the character code represents an accelerator or has a special meaning for the underlying windowing platform or (b) the FILTERIN especattribute is present for the event and the keycode or charcode associated with the event is not one of the keycodes or charcodes present in the value of that FILTERIN especattribute.

For gadgets that accept text input, only character codes that consist of a single graphic are included in the value of the gadget's VALUE elattribute. When a keyboard event occurs for which call back processing is enabled, the gadget's VALUE elattribute is not modified by the underlying windowing platform or the MWAPI. The application can modify the VALUE elattribute, by assigning a value explicitly.

* + 1. Focus Events

Focus events are focus, unfocus and change events.

A focus event, indicated by an etype value equal to "FOCUS", occurs when, as a result of user action, a window or gadget has received focus. A single action can produce a focus event for both a window and a gadget.

For a window, an unfocus event, indicated by an etype value equal to "UNFOCUS", occurs when, as a result of user action, a window is about to lose focus to another window defined in ^$WINDOW. If a window W1 which is defined in ^$WINDOW has focus, then focus changes to one or more windows not defined in ^$WINDOW, and then focus changes to a window W2 which is defined in ^$WINDOW, an unfocus event occurs for W1 if W1 is a different window than W2 and no unfocus event occurs for W1 if W1 is a the same window as W2.

For a gadget, an unfocus event occurs when the gadget is about to lose focus to another gadget defined for the same window. If a gadget G in window W has focus, then focus changes to one or more other windows, and then focus changes to window W, an unfocus event occurs for gadget G when focus changes to window W but not to gadget G.

A change event, indicated by an etype value equal to "CHANGE", occurs for a gadget instead of an unfocus event if (a) change events are enabled for the gadget and (b) the value of the gadget's CHANGED elattribute is equal to TRUE. If both change and unfocus events are enabled for a gadget and a change event occurs, then an unfocus event does not occur.

If, during call back processing for a change or unfocus event, an application removes the OK einfoattribute from ^$EVENT, the request to change focus is denied. If the action which produced the change or unfocus event also would produce a subsequent focus event, that focus event does not occur. In addition, any deferred events except timer events are discarded and, to the extent possible, visual and other effects of the attempt to change focus are prevented or reversed.

If an action yields both an unfocus or change event and a focus event, the unfocus or change event precedes the focus event.

* + 1. Select Events

Select events are select and unselect events.

A select event, indicated by an etype value equal to "SELECT" occurs when a user selects a gadget or choice.

For a check box, a select event occurs when the check box is selected.

For a list box, list button, list entry, long list box, or table gadget, a select event occurs for each choice that is selected. Multiple select events may occur as a result of a single user action.

For a menu item, a select event occurs when a choice is selected.

For a push button, a select event occurs when the push button is pushed.

For a radio button set, a select event occurs when a choice is selected. If a choice was previously selected, an unselect event precedes it indicating that a choice was implicitly deselected.

For a scroll gadget, a select event occurs when user action causes the scroll gadget's VALUE elattribute to change.

An unselect event, indicated by an etype value equal to "DESELECT" occurs when a user unselects a gadget or choice.

For a check box, an unselect event occurs when the check box is unselected.

For a list box, list button, list entry, long list box, or table gadget, an unselect event occurs for each choice that is unselected. For a list entry box gadget, an unselect event also occurs when a choice is selected and the user modifies the value of the list entry box. Multiple unselect events may occur as a result of a single user action.

* + 1. Long List Box Events

Long list box events (i.e. go bottom, go down, go down big, go top, go up, go up big events) occur when the user selects a movement control in a long list box gadget or uses a platform-specific key combination. If a long list box event is not enabled when a long list box gadget is created, the corresponding movement control may be absent from the long list box and it may not be possible to add it subsequently.

A go bottom event, indicated by an etype value equal to "GOBOTTOM", occurs for a gadget when the user selects the gadget's go bottom movement control.

A go down event, indicated by an etype value equal to "GODOWN", occurs for a gadget when the user selects the gadget's go down movement control.

A go down big event, indicated by an etype value equal to "GODOWNBIG", occurs for a gadget when the user selects the gadget's go down big movement control.

A go top event, indicated by an etype value equal to "GOTOP", occurs for a gadget when the user selects the gadget's go top movement control.

A go up event, indicated by an etype value equal to "GOUP", occurs for a gadget when the user selects the gadget's go up movement control.

A go up big event, indicated by an etype value equal to "GOUPBIG", occurs for a gadget when the user selects the gadget's go up big movement control.

If the event is caused by keyboard action the KEY einfoattribute contains the modifierkeys, if any, applying to the keyboard action, or emptystring. If the event is caused by pointer action the KEY einfoattribute is not present.

* + 1. Help Events

A help event, indicated by an etype value equal to "HELP", occurs when a window created via the MWAPI has focus and the user requests help by pressing a designated key or through some other means specified by the underlying windowing platform or by the MWAPI implementation. If a gadget for which help events are enabled has focus, then a help event occurs for that gadget. Otherwise, a help event occurs for the gadget's parent window.

* + 1. Timer Events

A timer event, indicated by an etype value equal to "TIMER", occurs when an interval timer expires.

* + 1. Data Exchange Events

Data Exchange events are associated with Data Exchange links. Unlike other MWAPI events these events are not disabled when the event owner (window or gadget) or any ancestor has its ACTIVE attribute set FALSE. Nor are the events deferred by a nested ESTART.

LINKOPEN occurs for an application window after another application establishes a link with that window. During callback processing for this event, the M process can remove the OK einfoattribute from ^$EVENT. In this case, the link is terminated when the callback completes.

LINKCLOSE occurs for an application window of a server when the link is terminated by the client. It occurs for the gadget of a client when the link is terminated by the server.

LINKACTION is caused when a request to perform an action on the link either completes or is not honoured within the interval defined by the gadget's LINKTIMEOUT attribute.

LINKNOTIFY is caused when a gadget's LINKMODE has the value NOTIFY, and the item defined by the gadget's LINKSOURCE and LINKITEM changes.

LINKUPDATE is caused when a gadget's LINKMODE has the value AUTOMATIC, and the item defined by the gadget's LINKSOURCE and LINKITEM changes. The event is caused after the gadget's VALUE elattribute is changed.

LINKPOKE is caused for a gadget when a client application is about to update the data source item which this gadget represents. During callback processing for this event, the M process can remove the OK einfoattribute from ^$EVENT, in which case the update is denied. Otherwise the new value will be assigned to the gadget immediately after the callback completes and before any other callback is processed.

LINKREQUEST is caused for a gadget when a client application requests the data source item which this gadget represents. During callback processing for this event, the M process can remove the OK einfoattribute from ^$EVENT, in which case the request is denied.

LINKEXECUTE is caused when a window acting as a data source is asked to execute a command. During callback processing for this event, the M process can remove the OK einfoattribute from $EVENT. In this case, the application requesting execution of the command is informed that the request failed.

* + 1. Build Mode Events

Build mode events are identified by a betype value.

For a build-mode window all the normal window state, pointer, keyboard, focus and help events can occur for the window. The pointer events PDOWN, PDRAG, PMOVE and PUP can occur while the pointer is over the window or is over a gadget on the window.

Additional build mode events can occur as follows.

A gadget-select event, indicated by a betype value equal to "GSELECT", occurs when the user selects a gadget, or when a selected gadget becomes the principal selection, or when a selected gadget loses the status of principal selection.

A gadget-unselect event, indicated by a betype value equal to "GDESELECT", occurs when the user unselects a gadget.

A gadget-resize event, indicated by a betype value equal to "GRESIZE", occurs when the user resizes a gadget.

A gadget-move event, indicated by a betype value equal to "GMOVE", occurs when the user moves a gadget. It does not occur when a gadget’s origin moves as a result of the user resizing it.

A menu-command event, indicated by a betype value equal to "MCOMMAND", occurs when the user selects a menu choice.

If a track event, indicated by a betype value equal to "TRACK", is defined and enabled, dragging the pointer on the window will cause a tracking rectangle to be displayed. One corner of the tracking rectangle is anchored at the point where the drag starts, and the diagonally-opposite corner tracks the pointer, which is constrained to the window’s viewport. If the SNAPPOS gridattr is TRUE the anchor point is adjusted to the nearest grid point. If the SNAPSIZE gridattr is TRUE the tracking point is adjusted to the nearest grid point. Tracking can be canceled by means of a platform-defined key sequence. Otherwise, the track event occurs when the user releases the pointer button after dragging the rectangle.

* + 1. Applicability of Event Types

The applicability of etypes to window, element, and choice types is specified by Table 4. A "Yes" entry indicates that the occurrence of the event type with the window, element, or choice type is defined. An empty entry indicates that the occurrence of the event type with the window, element, or choice type is reserved. implementationattribute values are applicable to all window, element and choice types.

| **Table 4: Events****Events** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | **Application**  **Window** | **M Term**  **Window** | **Check**  **Box** | **Document** | **Generic**  **Box** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CHANGE |  |  | Yes | Yes |  |
| CHAR | Yes |  |  | Yes | Yes |
| CLICK | Yes |  |  |  | Yes |
| CLOSE | Yes |  |  |  |  |
| DBLCLICK | Yes |  |  |  | Yes |
| DESELECT |  |  | Yes |  |  |
| FOCUS | Yes |  | Yes | Yes | Yes |
| GOBOTTOM |  |  |  |  |  |
| GODOWN |  |  |  |  |  |
| GODOWNBIG |  |  |  |  |  |
| GOTOP |  |  |  |  |  |
| GOUP |  |  |  |  |  |
| GOUPBIG |  |  |  |  |  |
| HELP | Yes |  | Yes | Yes | Yes |
| KEYDOWN | Yes |  |  | Yes | Yes |
| KEYUP | Yes |  |  | Yes | Yes |
| LINKACTION |  |  |  | Yes |  |
| LINKCLOSE | Yes |  |  | Yes |  |
| LINKEXECUTE | Yes |  |  |  |  |
| LINKNOTIFY |  |  |  | Yes |  |
| LINKOPEN | Yes |  |  |  |  |
| LINKPOKE |  |  |  | Yes |  |
| LINKREQUEST |  |  |  | Yes |  |
| LINKUPDATE |  |  |  | Yes |  |
| MAX | Yes |  |  |  |  |
| MIN | Yes |  |  |  |  |
| MOVE | Yes |  |  |  |  |
| PDOWN | Yes |  |  |  | Yes |
| PDRAG | Yes |  |  |  | Yes |
| PMOVE | Yes |  |  |  | Yes |
| PUP | Yes |  |  |  | Yes |
| RESIZE | Yes |  |  |  |  |
| RESTORE | Yes |  |  |  |  |
| SELECT |  |  | Yes |  |  |
| TIMER |  |  |  |  |  |
| UNFOCUS | Yes |  | Yes | Yes | Yes |

| **Table 4: Events (Continued)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | **Group**  **Frame** | **Label** | **List**  **Box** | **List**  **Button** | **List**  **Entry**  **Box** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CHANGE |  |  | Yes | Yes | Yes |
| CHAR |  |  |  |  | Yes |
| CLICK |  |  |  |  |  |
| CLOSE |  |  |  |  |  |
| DBLCLICK |  |  | Yes |  | Yes |
| DESELECT |  |  | Yes | Yes | Yes |
| FOCUS |  |  | Yes | Yes | Yes |
| GOBOTTOM |  |  |  |  |  |
| GODOWN |  |  |  |  |  |
| GODOWNBIG |  |  |  |  |  |
| GOTOP |  |  |  |  |  |
| GOUP |  |  |  |  |  |
| GOUPBIG |  |  |  |  |  |
| HELP |  |  | Yes | Yes | Yes |
| KEYDOWN |  |  |  |  | Yes |
| KEYUP |  |  |  |  | Yes |
| LINKACTION |  |  |  |  |  |
| LINKCLOSE |  |  |  |  |  |
| LINKEXECUTE |  |  |  |  |  |
| LINKNOTIFY |  |  |  |  |  |
| LINKOPEN |  |  |  |  |  |
| LINKPOKE |  |  |  |  |  |
| LINKREQUEST |  |  |  |  |  |
| LINKUPDATE |  |  |  |  |  |
| MAX |  |  |  |  |  |
| MIN |  |  |  |  |  |
| MOVE |  |  |  |  |  |
| PDOWN |  |  |  |  |  |
| PDRAG |  |  |  |  |  |
| PMOVE |  |  |  |  |  |
| PUP |  |  |  |  |  |
| RESIZE |  |  |  |  |  |
| RESTORE |  |  |  |  |  |
| SELECT |  |  | Yes | Yes | Yes |
| TIMER |  |  |  |  |  |
| UNFOCUS |  |  | Yes | Yes | Yes |

| **Table 4: Events (Continued)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | **Long**  **List**  **Box** | **Menu** | **Menu**  **Choice** | **Push**  **Button** | **Radio**  **Button**  **Set** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CHANGE | Yes |  |  |  | Yes |
| CHAR |  |  |  |  |  |
| CLICK |  |  |  |  |  |
| CLOSE |  |  |  |  |  |
| DBLCLICK | Yes |  |  |  |  |
| DESELECT | Yes |  |  |  | Yes |
| FOCUS | Yes |  |  | Yes | Yes |
| GOBOTTOM | Yes |  |  |  |  |
| GODOWN | Yes |  |  |  |  |
| GODOWNBIG | Yes |  |  |  |  |
| GOTOP | Yes |  |  |  |  |
| GOUP | Yes |  |  |  |  |
| GOUPBIG | Yes |  |  |  |  |
| HELP | Yes |  |  | Yes | Yes |
| KEYDOWN |  |  |  |  |  |
| KEYUP |  |  |  |  |  |
| LINKACTION |  |  |  |  |  |
| LINKCLOSE |  |  |  |  |  |
| LINKEXECUTE |  |  |  |  |  |
| LINKNOTIFY |  |  |  |  |  |
| LINKOPEN |  |  |  |  |  |
| LINKPOKE |  |  |  |  |  |
| LINKREQUEST |  |  |  |  |  |
| LINKUPDATE |  |  |  |  |  |
| MAX |  |  |  |  |  |
| MIN |  |  |  |  |  |
| MOVE |  |  |  |  |  |
| PDOWN |  |  |  |  |  |
| PDRAG |  |  |  |  |  |
| PMOVE |  |  |  |  |  |
| PUP |  |  |  |  |  |
| RESIZE |  |  |  |  |  |
| RESTORE |  |  |  |  |  |
| SELECT | Yes |  | Yes | Yes | Yes |
| TIMER |  |  |  |  |  |
| UNFOCUS | Yes |  |  | Yes | Yes |

| **Table 4: Events (Continued)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  | **Scroll**  **Bar** | **Symbol** | **Table** | **Text** | **Timer** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CHANGE |  |  | Yes | Yes |  |
| CHAR |  |  |  | Yes |  |
| CLICK |  |  | Yes |  |  |
| CLOSE |  |  |  |  |  |
| DBLCLICK |  |  | Yes |  |  |
| DESELECT |  |  | Yes |  |  |
| FOCUS |  |  | Yes | Yes |  |
| GOBOTTOM |  |  |  |  |  |
| GODOWN |  |  |  |  |  |
| GODOWNBIG |  |  |  |  |  |
| GOTOP |  |  |  |  |  |
| GOUP |  |  |  |  |  |
| GOUPBIG |  |  |  |  |  |
| HELP | Yes |  | Yes | Yes |  |
| KEYDOWN |  |  |  | Yes |  |
| KEYUP |  |  |  | Yes |  |
| LINKACTION |  |  |  | Yes |  |
| LINKCLOSE |  |  |  | Yes |  |
| LINKEXECUTE |  |  |  |  |  |
| LINKNOTIFY |  |  |  | Yes |  |
| LINKOPEN |  |  |  |  |  |
| LINKPOKE |  |  |  | Yes |  |
| LINKREQUEST |  |  |  | Yes |  |
| LINKUPDATE |  |  |  | Yes |  |
| MAX |  |  |  |  |  |
| MIN |  |  |  |  |  |
| MOVE |  |  |  |  |  |
| PDOWN |  |  | Yes |  |  |
| PDRAG | Yes |  | Yes |  |  |
| PMOVE |  |  | Yes |  |  |
| PUP |  |  | Yes |  |  |
| RESIZE |  |  |  |  |  |
| RESTORE |  |  |  |  |  |
| SELECT | Yes |  | Yes |  |  |
| TIMER |  |  |  |  | Yes |
| UNFOCUS |  |  | Yes | Yes |  |

* 1. Event Specification Attributes

Event specifications are defined by especs which consist of lists of subscripts.

espec ::= L expr

For windows, an espec has the form

wname , ekeyword , etype

where

|  |  |  |
| --- | --- | --- |
| ekeyword ::= expr V | BEVENT EVENT |  |

For gadgets, an espec has the form

wname , gkeyword , gname , ekeyword , etype

For timers, an espec has the form

wname , tkeyword , tname , ekeyword , etype

For menu choices, an espec has the form

wname , mkeyword , mname , ckeyword , item , ekeyword , etype

All other forms of espec are reserved.

Event specification attributes, identified by especattributes, determine when events and call backs occur.

|  |  |
| --- | --- |
| especattribute ::= expr V | ENABLE  FILTERIN  implementationattribute |

Event specification attributes are specified as:

^$W[INDOW] ( espec , especattribute )

* + 1. ENABLE especattribute

**Definition** An event identified by etype is enabled for a window, gadget, timer, or menu choice if, when the event occurs, there is an espec defined for the window, gadget, timer or menu choice which has a valid doargument for its value and the corresponding ENABLE especattribute for the espec has a TRUE value. If the ENABLE especattribute is FALSE, the event is not enabled for the window, gadget, timer or menu choice.

**Domain** truefalse

**Access** Create, Modify, Reference

**Default** TRUE

* + 1. FILTERIN especattribute

**Definition** FILTERIN specifies, for certain events, additional criteria that must be TRUE for an event to occur.

**Domain** For a key down or key up event:

L keycode

For a character event:

L charcode

For a pointer event:

L pbuttoncode

**Access** Create, Modify, Reference

**Default** None

* 1. Processing Events

If event processing is activated, then for each enabled event of type E that occurs, call back processing occurs by execution of an implicit DO command of the form

DO @doargument

where doargument is the value of the EVENT wattribute, elattribute, or cattribute for a window, element, or choice, respectively for etype equal to E, or for a build mode window where doargument is the value of the BEVENT wattribute for betype equal to E..

For each ESTART command, call back processing is carried out for events in the order of their occurrence. If an enabled event occurs while call back processing is not activated, call back processing for it may be deferred until call back processing is activated. While there is no explicit limit on the number of deferred events, deferred events may be lost if not processed quickly.

Call back processing does not occur as a result of modification of ^$DISPLAY or ^$WINDOW by the application.

For close, data key up, and data key down events, if the event is enabled, the normal actions of the underlying windowing platform for that event are inhibited; if the event is not enabled, the normal actions of the underlying windowing platform for that event occur. For other events, the normal actions of the underlying windowing platform occur for the event, whether or not the event is enabled.

* 1. Event Commands

* + 1. ESTART

|  |  |
| --- | --- |
| ESTA [ RT ] postcond | [ SP ]  SP timeout |

The M command ESTART activates event processing. Event processing remains activated until the termination of execution of the ESTART command, except that event processing is implicitly deactivated at the initiation of call back processing for each event. At the conclusion of call back processing for each event, event processing is implicitly reactivated and all einfoattribute nodes in ^$EVENT become undefined.

During call back processing for an event E, event processing can be explicitly activated by executing a nested ESTART command. This causes call back processing for event E to be suspended, and causes the values of all einfoattribute nodes in ^$EVENT to be stacked and to become undefined. When execution of this nested ESTART command is terminated, the values of the einfoattribute nodes that it stacked are restored to ^$EVENT.

During call back processing for event E and prior to execution of a nested ESTART command, the FOCUS dattribute for the display on which the window associated with event E appears must be assigned a value which identifies a window W which has a MODAL wattribute value equal to "APPLICATION". If FOCUS is not assigned such a value, an error occurs with an ecode = "M55". If FOCUS is assigned such a value, the nested ESTART command activates event processing for (a) any enabled events for window W and (b) any enabled timer events for all other windows; processing of all other enabled events is deferred.

The execution of an ESTART command is terminated if (a) an ESTOP command is executed during call back processing for that ESTART command or (b) if timeout is present, the timeout expires. If the timeout period expires during callback processing for an event, the ESTART command is terminated at the conclusion of processing for that event.

When execution of an ESTART command is terminated, execution continues with the command following that ESTART command.

If timeout is present, $TEST indicates the condition that caused event processing to terminate. If event processing was terminated with an ESTOP command, the value of $TEST is 1. If event processing was not terminated with an ESTOP command, the value of $TEST is 0. If timeout is not present, execution of ESTART does not change $TEST.

* + 1. ESTOP

ESTO [ P ] postcond [ SP ]

The M command ESTOP implicitly performs the number of M QUIT commands necessary to return to the execution level of the most recently executed ESTART command and then terminates that ESTART command. If event processing is not activated, execution of an ESTOP command has no effect.

* + 1. ETRIGGER

ETR [ IGGER ] postcond SP especref [ : einforef ]

The M command ETRIGGER causes an event to occur. If this event is enabled, call back processing for it will occur subsequently.

especref ::= expr V ^$W [ INDOW ] ( espec )

|  |  |
| --- | --- |
| einforef ::= | glvn  @ expratom V einforef |

The event occurs for the window, element, or menu choice specified by the especref. If the window, element, or menu choice specified by the especref does not exist, an error condition occurs with an ecode = "M48" and $EREF containing a value that indicates the non-existent window, element, or menu choice.

The event that occurs is specified by the etype component of the especref. If the especref does not contain a valid etype, an error condition occurs with an ecode = "M47".

No additional events occur as consequence of an event that occurs as a result of execution of an ETRIGGER command. For example, executing an ETRIGGER command that causes a focus event to occur does not automatically cause an unfocus or change event to occur.

When call back processing occurs for the event, the following einfoattribute nodes are provided automatically in ^$EVENT:

For all events, CLASS, SEQUENCE, TYPE and WINDOW.

When an event occurs for an element, ELEMENT.

When an event occurs for a menu choice, CHOICE and ELEMENT.

For change and unfocus events, OK.

Other einfoattribute nodes can be provided by the application via the einforef argument. If einforef is present, then for each immediately descendant subscript of einforef which matches an einfoattribute that (a) is defined for the event type specified in the especref and (b) is not provided automatically in ^$EVENT, as described above, the value of the ^$EVENT einfoattribute is set to the value of the corresponding einforef node.

* 1. Event Information

During call back processing for an event, the ssvn ^$EVENT contains event information attributes, identified by einfoattributes, in the form

^$E[VENT] ( einfoattribute )

which provide information about the event.

|  |  |
| --- | --- |
| einfoattribute ::= expr V | CHOICE  CLASS  COL  ELEMENT GROUP  KEY LINKCOMMAND LINKID LINKITEM LINKREASON LINKSOURCE LINKSTATUS  NEXTFOCUS  OK  PBUTTON  PPOS  PRIORFOCUS PRIORPOS PSELECT  PSTATE  ROW  SEQUENCE TRACKPOS TRACKSIZE  TYPE  WINDOW  implementationattribute |

* + 1. CHOICE einfoattribute

**Definition** CHOICE identifies the choice associated with a select event for an element.

**Domain** item

**Access** Reference

**Default** Not Applicable.

* + 1. CLASS einfoattribute

**Definition** CLASS specifies the event class to which the event belongs. For Data Exchange events, CLASS is equal to "LINK". For build mode events, CLASS is equal to “BUILD”For all other MWAPI events, CLASS is equal to "WINDOW".

**Domain** chars

**Access** Reference

**Default** Not Applicable.

* + 1. COL einfoattribute

**Definition** COL specifies the column associated with a select or pointer event for a gadget that has the COL attribute.

**Domain** citem

**Access** Reference

**Default** Not Applicable.

* + 1. ELEMENT einfoattribute

**Definition** If the event is associated with an element, ELEMENT specifies that element's type and ename. If the event is not associated with an element, ELEMENT is not defined.

**Domain** For a gadget

G,ename

For a menu

M,ename

For a timer

T,ename

**Access** Reference

**Default** Not Applicable.

* + 1. GROUP einfoattribute

**Definition** GROUP allows related build-mode events to be recognized. A single user action may cause multiple selection and/or deselection events, or multiple move events. All events caused by a single action have the same value of GROUP. The actual value is arbitrary.

**Domain** intlit

**Access** Reference

**Default** Not Applicable.

* + 1. KEY einfoattribute

**Definition** KEY identifies the key sequence associated with a keyboard event.

**Domain** For a key up or key down event:

keycode

For a character event:

charcode

**Access** Reference

**Default** Not Applicable.

* + 1. LINKCOMMAND einfoattribute

**Definition** Defines the command string associated with a LINKEXECUTE event.

**Domain** chars

**Access** Reference

**Default** Not Applicable.

* + 1. LINKID einfoattribute

**Definition** Identifies the link associated with a data exchange event.

**Domain** delinkid

**Access** Reference

**Default** Not Applicable.

* + 1. LINKITEM einfoattribute

**Definition** Identifies the item associated with a data exchange event.

**Domain** deitem

**Access** Reference

**Default** Not Applicable.

* + 1. LINKREASON einfoattribute

**Definition** Identifies the platform-specific reason for the failure of an action requested on a data exchange link.

**Domain** chars

**Access** Reference

**Default** Not Applicable.

* + 1. LINKSOURCE einfoattribute

**Definition** Identifies the data source associated with a data exchange event.

**Domain** detopic

**Access** Reference

**Default** Not Applicable.

* + 1. LINKSTATUS einfoattribute

**Definition** Identifies the outcome of an action requested on a data exchange link.

**Domain** destatus

**Access** Reference

**Default** Not Applicable.

* + 1. NEXTFOCUS einfoattribute

**Definition** During call back processing for an unfocus or change event, NEXTFOCUS identifies the window or gadget for which a related focus event will occur. If no related focus event will occur for a window or gadget that is defined in ^$WINDOW, NEXTFOCUS is equal to emptystring.

**Domain** focusloc

**Access** Reference

**Default** Not Applicable.

* + 1. OK einfoattribute

**Definition** OK specifies whether the event should be completed normally. If OK is present, processing for the event completes normally. If OK is not defined, processing for the event is terminated in an event-specific fashion and the normal actions of the underlying windowing platform for that event are inhibited in an event-specific fashion.

An application cannot assign a value to the OK einfoattribute but it can remove the OK einfoattribute using the M KILL command.

**Domain** emptystring

**Access** Modify, Reference

**Default** emptystring

* + 1. PBUTTON einfoattribute

**Definition** PBUTTON identifies the pointer button and modifier keys, if any, associated with a pointer event.

**Domain** pbuttoncode

**Access** Reference

**Default** Not Applicable.

* + 1. PPOS einfoattribute

**Definition** PPOS identifies the pointer position. If the event is associated with a gadget, the position is measured with respect to the gadget's origin and is expressed in the unit of measure specified by the gadget's UNITS elattribute. Otherwise, the event is associated with a window and the position is measured with respect to the origin of the application area and is expressed in the unit of measure specified by the window's UNITS wattribute.

**Domain** pos

**Access** Reference

**Default** Not Applicable.

* + 1. PRIORFOCUS einfoattribute

**Definition** During call back processing for a focus event, PRIORFOCUS identifies the window or gadget for which a related unfocus or change event occurred. If no related unfocus or change event occurred for a window or gadget that is defined in ^$WINDOW, PRIORFOCUS is equal to emptystring.

**Domain** focusloc

**Access** Reference

**Default** Not Applicable.

* + 1. PRIORPOS einfoattribute

**Definition** PRIORPOS identifies the position of the gadget before it was moved.

**Domain** pos

**Access** Reference

**Default** Not Applicable.

* + 1. PSELECT einfoattribute

**Definition** PSELECT is TRUE for a GDESELECT event occurring for a gadget that was the principal selection at the time it was unselected. It is TRUE for a GSELECT occurring for a gadget that is becoming the principal selection. Otherwise it is FALSE.

**Domain** truefalse

**Access** Reference

**Default** Not Applicable.

* + 1. PSTATE einfoattribute

**Definition** PSTATE identifies any pointer buttons that were pressed when a pointer event occurred. If no buttons were pressed, PSTATE is equal to emptystring.

**Domain** L pbuttoncode

**Access** Reference

**Default** Not Applicable.

* + 1. ROW einfoattribute

**Definition** ROW specifies the row associated with a select or pointer event for a gadget that has the ROW attribute.

**Domain** ritem

**Access** Reference

**Default** Not Applicable.

* + 1. SEQUENCE einfoattribute

**Definition** SEQUENCE uniquely identifies each event. The value of SEQUENCE is the number of events for which call back processing has been initiated for a process.

**Domain** intlit

**Access** Reference

**Default** Not Applicable.

* + 1. TRACKPOS einfoattribute

**Definition** TRACKPOS identifies the position of the top left corner of the tracking rectangle associated with the event.

**Domain** pos

**Access** Reference

**Default** Not Applicable.

* + 1. TRACKSIZE einfoattribute

**Definition** TRACKSIZE identifies the size of the tracking rectangle associated with the event.

**Domain** size

**Access** Reference

**Default** Not Applicable.

* + 1. TYPE einfoattribute

**Definition** TYPE identifies the type of event.

**Domain** etype

**Access** Reference

**Default** Not Applicable.

* + 1. WINDOW einfoattribute

**Definition** If the event is associated with a window, WINDOW specifies that window's wname. If the event is associated with an element, WINDOW specifies the wname of the element's parent. If the event is not associated with a window or element, WINDOW is not defined.

**Domain** wname

**Access** Reference

**Default** Not Applicable.

* + 1. Applicability of Event Information Attributes

The applicability of einfoattribute nodes to etypes and betypes is specified by Table 5. A "Yes" entry indicates that the einfoattribute is defined for the etype or betype. An empty entry indicates that the einfoattribute is reserved for the etype or betype. implementationattribute values are applicable to all etypes and betypes.

| **Table 5: Event Information****Event Information** | | | | |
| --- | --- | --- | --- | --- |
|  | **CHOICE** | **CLASS** | **COL** | **ELEMENT** |
| CHANGE |  | Yes |  | Yes |
| CHAR |  | Yes |  | Yes\* |
| CLICK |  | Yes | Yes | Yes\* |
| CLOSE |  | Yes |  |  |
| DBLCLICK |  | Yes | Yes | Yes\* |
| DESELECT | Yes | Yes | Yes | Yes |
| FOCUS |  | Yes |  | Yes\* |
| GDESELECT |  | Yes |  | Yes |
| GMOVE |  | Yes |  | Yes |
| GOBOTTOM |  | Yes |  | Yes |
| GODOWN |  | Yes |  | Yes |
| GODOWNBIG |  | Yes |  | Yes |
| GOTOP |  | Yes |  | Yes |
| GOUP |  | Yes |  | Yes |
| GOUPBIG |  | Yes |  | Yes |
| GRESIZE |  | Yes |  | Yes |
| GSELECT |  | Yes |  | Yes |
| HELP |  | Yes |  | Yes\* |
| KEYDOWN |  | Yes |  | Yes\* |
| KEYUP |  | Yes |  | Yes\* |
| LINKACTION |  | Yes |  | Yes |

\* The ELEMENT einfoattribute is defined if the event occurs for an element and is undefined otherwise.

|  |
| --- |
| **Table 5: Event Information****Event Information (Continued)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LINKCLOSE |  | Yes |  | Yes\* |
| LINKEXECUTE |  | Yes |  |  |
| LINKNOTIFY |  | Yes |  | Yes |
| LINKOPEN |  | Yes |  |  |
| LINKPOKE |  | Yes |  | Yes |
| LINKREQUEST |  | Yes |  | Yes |
| LINKUPDATE |  | Yes |  | Yes |
| MAX |  | Yes |  |  |
| MCOMMAND | Yes | Yes |  | Yes |
| MIN |  | Yes |  |  |
| MOVE |  | Yes |  |  |
| PDOWN |  | Yes | Yes | Yes\* |
| PDRAG |  | Yes |  | Yes\* |
| PMOVE |  | Yes |  | Yes\* |
| PUP |  | Yes | Yes | Yes\* |
| RESIZE |  | Yes |  |  |
| RESTORE |  | Yes |  |  |
| SELECT | Yes | Yes | Yes | Yes |
| TIMER |  | Yes |  | Yes |
| TRACK |  | Yes |  |  |
| UNFOCUS |  | Yes |  | Yes\* |

\* The ELEMENT einfoattribute is defined if the event occurs for an element and is undefined otherwise.

| **Table 5: Event Information (Continued)** | | | | |
| --- | --- | --- | --- | --- |
|  | **GROUP** | **KEY** | **LINK-COMMAND** | **LINKID** |
| CHANGE |  |  |  |  |
| CHAR |  | Yes |  |  |
| CLICK |  |  |  |  |
| CLOSE |  |  |  |  |
| DBLCLICK |  |  |  |  |
| DESELECT |  |  |  |  |
| FOCUS |  |  |  |  |
| GDESELECT | Yes |  |  |  |
| GMOVE | Yes |  |  |  |
| GOBOTTOM |  |  |  |  |
| GODOWN |  |  |  |  |
| GODOWNBIG |  |  |  |  |
| GOTOP |  |  |  |  |
| GOUP |  |  |  |  |
| GOUPBIG |  |  |  |  |
| GRESIZE |  |  |  |  |
| GSELECT | Yes |  |  |  |
| HELP |  |  |  |  |
| KEYDOWN |  | Yes |  |  |
| KEYUP |  | Yes |  |  |
| LINKACTION |  |  |  | Yes |

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LINKCLOSE |  |  |  | Yes |
| LINKEXECUTE |  |  | Yes | Yes |
| LINKNOTIFY |  |  |  | Yes |
| LINKOPEN |  |  |  | Yes |
| LINKPOKE |  |  |  | Yes |
| LINKREQUEST |  |  |  | Yes |
| LINKUPDATE |  |  |  | Yes |
| MAX |  |  |  |  |
| MCOMMAND |  |  |  |  |
| MIN |  |  |  |  |
| MOVE |  |  |  |  |
| PDOWN |  |  |  |  |
| PDRAG |  |  |  |  |
| PMOVE |  |  |  |  |
| PUP |  |  |  |  |
| RESIZE |  |  |  |  |
| RESTORE |  |  |  |  |
| SELECT |  |  |  |  |
| TIMER |  |  |  |  |
| TRACK |  |  |  |  |
| UNFOCUS |  |  |  |  |

| **Table 5: Event Information (Continued)** | | | | |
| --- | --- | --- | --- | --- |
|  | **LINKITEM** | **LINK-REASON** | **LINK-SOURCE** | **LINK-STATUS** |
| CHANGE |  |  |  |  |
| CHAR |  |  |  |  |
| CLICK |  |  |  |  |
| CLOSE |  |  |  |  |
| DBLCLICK |  |  |  |  |
| DESELECT |  |  |  |  |
| FOCUS |  |  |  |  |
| GDESELECT |  |  |  |  |
| GMOVE |  |  |  |  |
| GOBOTTOM |  |  |  |  |
| GODOWN |  |  |  |  |
| GODOWNBIG |  |  |  |  |
| GOTOP |  |  |  |  |
| GOUP |  |  |  |  |
| GOUPBIG |  |  |  |  |
| GRESIZE |  |  |  |  |
| GSELECT |  |  |  |  |
| HELP |  |  |  |  |
| KEYDOWN |  |  |  |  |
| KEYUP |  |  |  |  |
| LINKACTION | Yes | Yes | Yes | Yes |

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LINKCLOSE |  |  | Yes |  |
| LINKEXECUTE |  |  | Yes |  |
| LINKNOTIFY | Yes |  | Yes |  |
| LINKOPEN |  |  | Yes |  |
| LINKPOKE |  |  | Yes |  |
| LINKREQUEST |  |  | Yes |  |
| LINKUPDATE | Yes |  | Yes |  |
| MAX |  |  |  |  |
| MCOMMAND |  |  |  |  |
| MIN |  |  |  |  |
| MOVE |  |  |  |  |
| PDOWN |  |  |  |  |
| PDRAG |  |  |  |  |
| PMOVE |  |  |  |  |
| PUP |  |  |  |  |
| RESIZE |  |  |  |  |
| RESTORE |  |  |  |  |
| SELECT |  |  |  |  |
| TIMER |  |  |  |  |
| TRACK |  |  |  |  |
| UNFOCUS |  |  |  |  |

| **Table 5: Event Information (Continued)** | | | | |
| --- | --- | --- | --- | --- |
|  | **NEXT-**  **FOCUS** | **OK** | **PBUTTON** | **PPOS** |
| CHANGE | Yes | Yes |  |  |
| CHAR |  |  |  |  |
| CLICK |  |  | Yes | Yes |
| CLOSE |  |  |  |  |
| DBLCLICK |  |  | Yes | Yes |
| DESELECT |  |  |  |  |
| FOCUS |  |  |  |  |
| GDESELECT |  |  | Yes+ |  |
| GMOVE |  |  | Yes |  |
| GOBOTTOM |  |  |  |  |
| GODOWN |  |  |  |  |
| GODOWNBIG |  |  |  |  |
| GOTOP |  |  |  |  |
| GOUP |  |  |  |  |
| GOUPBIG |  |  |  |  |
| GRESIZE |  |  |  |  |
| GSELECT |  |  | Yes+ |  |
| HELP |  |  |  |  |
| KEYDOWN |  |  |  |  |
| KEYUP |  |  |  |  |
| LINKACTION |  |  |  |  |

+ The PPOS einfoattribute is undefined for selection or deselection not performed with the pointer.

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LINKCLOSE |  |  |  |  |
| LINKEXECUTE |  | Yes |  |  |
| LINKNOTIFY |  |  |  |  |
| LINKOPEN |  | Yes |  |  |
| LINKPOKE |  | Yes |  |  |
| LINKREQUEST |  | Yes |  |  |
| LINKUPDATE |  |  |  |  |
| MAX |  |  |  |  |
| MCOMMAND |  |  |  |  |
| MIN |  |  |  |  |
| MOVE |  |  |  |  |
| PDOWN |  |  | Yes | Yes |
| PDRAG |  |  | Yes | Yes# |
| PMOVE |  |  | Yes | Yes |
| PUP |  |  | Yes | Yes |
| RESIZE |  |  |  |  |
| RESTORE |  |  |  |  |
| SELECT |  |  |  |  |
| TIMER |  |  |  |  |
| TRACK |  |  | Yes |  |
| UNFOCUS | Yes | Yes |  |  |

# The PPOS einfoattribute is undefined for PDRAG events associated with scroll gadgets.

| **Table 5: Event Information (Continued)** | | | | |
| --- | --- | --- | --- | --- |
|  | **PRIOR-**  **FOCUS** | **PRIORPOS** | **PSELECT** | **PSTATE** |
| CHANGE |  |  |  |  |
| CHAR |  |  |  |  |
| CLICK |  |  |  | Yes |
| CLOSE |  |  |  |  |
| DBLCLICK |  |  |  | Yes |
| DESELECT |  |  |  |  |
| FOCUS | Yes |  |  |  |
| GDESELECT |  |  | Yes | Yes+ |
| GMOVE |  | Yes |  | Yes |
| GOBOTTOM |  |  |  |  |
| GODOWN |  |  |  |  |
| GODOWNBIG |  |  |  |  |
| GOTOP |  |  |  |  |
| GOUP |  |  |  |  |
| GOUPBIG |  |  |  |  |
| GRESIZE |  |  |  |  |
| GSELECT |  |  | Yes | Yes+ |
| HELP |  |  |  |  |
| KEYDOWN |  |  |  |  |
| KEYUP |  |  |  |  |
| LINKACTION |  |  |  |  |

+ The PSTATE einfoattribute is undefined for selection or deselection not performed with the pointer.

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LINKCLOSE |  |  |  |  |
| LINKEXECUTE |  |  |  |  |
| LINKNOTIFY |  |  |  |  |
| LINKOPEN |  |  |  |  |
| LINKPOKE |  |  |  |  |
| LINKREQUEST |  |  |  |  |
| LINKUPDATE |  |  |  |  |
| MAX |  |  |  |  |
| MCOMMAND |  |  |  |  |
| MIN |  |  |  |  |
| MOVE |  |  |  |  |
| PDOWN |  |  |  | Yes |
| PDRAG |  |  |  | Yes |
| PMOVE |  |  |  | Yes |
| PUP |  |  |  | Yes |
| RESIZE |  |  |  |  |
| RESTORE |  |  |  |  |
| SELECT |  |  |  |  |
| TIMER |  |  |  |  |
| TRACK |  |  |  | Yes |
| UNFOCUS |  |  |  |  |

| **Table 5: Event Information (Continued)** | | | | |
| --- | --- | --- | --- | --- |
|  | **ROW** | **SEQUENCE** | **TRACKPOS** | **TRACKSIZE** |
| CHANGE |  | Yes |  |  |
| CHAR |  | Yes |  |  |
| CLICK | Yes | Yes |  |  |
| CLOSE |  | Yes |  |  |
| DBLCLICK | Yes | Yes |  |  |
| DESELECT | Yes | Yes |  |  |
| FOCUS |  | Yes |  |  |
| GDESELECT |  | Yes |  |  |
| GMOVE |  | Yes |  |  |
| GOBOTTOM |  | Yes |  |  |
| GODOWN |  | Yes |  |  |
| GODOWNBIG |  | Yes |  |  |
| GOTOP |  | Yes |  |  |
| GOUP |  | Yes |  |  |
| GOUPBIG |  | Yes |  |  |
| GRESIZE |  | Yes |  |  |
| GSELECT |  | Yes |  |  |
| HELP |  | Yes |  |  |
| KEYDOWN |  | Yes |  |  |
| KEYUP |  | Yes |  |  |
| LINKACTION |  | Yes |  |  |

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LINKCLOSE |  | Yes |  |  |
| LINKEXECUTE |  | Yes |  |  |
| LINKNOTIFY |  | Yes |  |  |
| LINKOPEN |  | Yes |  |  |
| LINKPOKE |  | Yes |  |  |
| LINKREQUEST |  | Yes |  |  |
| LINKUPDATE |  | Yes |  |  |
| MAX |  | Yes |  |  |
| MCOMMAND |  | Yes |  |  |
| MIN |  | Yes |  |  |
| MOVE |  | Yes |  |  |
| PDOWN | Yes | Yes |  |  |
| PDRAG |  | Yes |  |  |
| PMOVE |  | Yes |  |  |
| PUP | Yes | Yes |  |  |
| RESIZE |  | Yes |  |  |
| RESTORE |  | Yes |  |  |
| SELECT | Yes | Yes |  |  |
| TIMER |  | Yes |  |  |
| TRACK |  | Yes | Yes | Yes |
| UNFOCUS |  | Yes |  |  |

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |
| --- | --- | --- |
|  | **TYPE** | **WINDOW** |
| CHANGE | Yes | Yes |
| CHAR | Yes | Yes |
| CLICK | Yes | Yes |
| CLOSE | Yes | Yes |
| DBLCLICK | Yes | Yes |
| DESELECT | Yes | Yes |
| FOCUS | Yes | Yes |
| GDESELECT | Yes | Yes |
| GMOVE | Yes | Yes |
| GOBOTTOM | Yes | Yes |
| GODOWN | Yes | Yes |
| GODOWNBIG | Yes | Yes |
| GOTOP | Yes | Yes |
| GOUP | Yes | Yes |
| GOUPBIG | Yes | Yes |
| GRESIZE | Yes | Yes |
| GSELECT | Yes | Yes |
| HELP | Yes | Yes |
| KEYDOWN | Yes | Yes |
| KEYUP | Yes | Yes |
| LINKACTION | Yes | Yes |

|  |
| --- |
| **Table 5: Event Information (Continued)** |

|  |  |  |
| --- | --- | --- |
| LINKCLOSE | Yes | Yes |
| LINKEXECUTE | Yes | Yes |
| LINKNOTIFY | Yes | Yes |
| LINKOPEN | Yes | Yes |
| LINKPOKE | Yes | Yes |
| LINKREQUEST | Yes | Yes |
| LINKUPDATE | Yes | Yes |
| MAX | Yes | Yes |
| MCOMMAND | Yes | Yes |
| MIN | Yes | Yes |
| MOVE | Yes | Yes |
| PDOWN | Yes | Yes |
| PDRAG | Yes | Yes |
| PMOVE | Yes | Yes |
| PUP | Yes | Yes |
| RESIZE | Yes | Yes |
| RESTORE | Yes | Yes |
| SELECT | Yes | Yes |
| TIMER | Yes | Yes |
| TRACK | Yes | Yes |
| UNFOCUS | Yes | Yes |

# 

# Additional Functions and Special Variables

* 1. Functions

When the MWAPI is available, the following functions are added to M:

$WFONT

$WTFIT

$WTWIDTH

* + 1. $WFONT

$WFONT ( fontface , fontsize , fontstyle , unitspec )

$WFONT returns information about the font identified by fontface, fontsize, and fontstyle. The value returned has the form

fontheight , fontmaxchar , fontavgchar

where

fontheight ::= numlit

fontmaxchar ::= numlit

fontavgchar ::= numlit

fontheight is the height of the character cell plus external leading for the font. fontmaxchar is the maximum width of a single character for the font. fontavgchar is an average width of a single character for the font.

If unitspec is equal to "PIXEL", fontheight, fontmaxchar, and fontavgchar are measured in pixels. If unitspec is equal to "POINT", fontheight, fontmaxchar, and fontavgchar are measured in points. If any other value is specified for unitspec, an error condition occurs with an ecode = "M53".

If information is not available for the specified font, the return value is equal to the emptystring.

* + 1. $WTFIT

$WTFIT ( expr , numexpr , fontface , fontsize , fontstyle , unitspec )

$WTFIT returns a value equal to the number of characters in the text string expr that could be displayed fully, using an indicated font, in an area whose width is numexpr in the unit of measure specified by unitspec. wide. The indicated font is specified by fontface, fontsize and fontstyle.

For a string expr containing c characters, $WTFIT returns the following values:

If the entire string expr can be displayed in an area numexpr units wide, $WTFIT returns c.

If the first character in the string expr is too wide to be displayed fully in an area numexpr units wide, $WTFIT returns 0 (zero).

If the first m characters in the string expr can be displayed fully in an area numexpr units wide, but the first m+1 characters are too wide to be displayed in an area numexpr units wide, $WTFIT returns m.

If information is not available for the specified font, the return value is equal to the emptystring.

If unitspec is not equal to "PIXEL" or "POINT", an error condition occurs with an ecode = "M53".

* + 1. $WTWIDTH

$WTWIDTH ( expr , fontface , fontsize , fontstyle , unitspec )

$WTWIDTH returns a value equal to the width of the text string expr displayed using the font indicated by fontface, fontsize and fontstyle. The width is expressed in the unit of measure indicated by unitspec.

If information is not available for the specified font, the return value is equal to the emptystring.

If unitspec is not equal to "PIXEL" or "POINT", an error condition occurs with an ecode = "M53".

* 1. Special Variables

The following svn is added to M:

$PD[ISPLAY]

The intrinsic special variable $PDISPLAY indicates the principal display of an M process. The value of $PDISPLAY is a display.

# 

# Error Handling

Errors that occur during use of the MWAPI are handled in accordance with the error processing mechanisms specified for M.

If one or more MWAPI errors occurs, at least one error is reported. If one or more errors occurs during the execution of an M MERGE command that references ^$DISPLAY or ^$WINDOW as a glvn on the left hand side of the equal sign, the resulting state of ^$DISPLAY or ^$WINDOW is not defined.

* 1. Error Information

For certain MWAPI errors, an additional svn, $EREF, identifies the ssvn reference associated with the error.

$ER[EF]

The value provided by $EREF is a namevalue.

If the error involves a display attribute, $EREF has the form:

^$DISPLAY ( display , dattribute )

If the error involves a window attribute, $EREF has the form:

^$WINDOW ( wname , wattribute )

If the error involves a gadget attribute, $EREF has the form:

^$WINDOW ( wname , gkeyword , gname , elattribute )

If the error involves a menu attribute, $EREF has the form:

^$WINDOW ( wname , mkeyword , mname , elattribute )

If the error involves a timer attribute, $EREF has the form:

^$WINDOW ( wname , tkeyword , tname , elattribute )

If the error involves a column attribute, $EREF has the form:

^$WINDOW ( wname , gkeyword , gname , colkeyword , citem , colattribute )

If the error involves a row attribute, $EREF has the form:

^$WINDOW ( wname , gkeyword , gname , rowkeyword , ritem , rowattribute )

If the error involves a choice attribute for a table gadget, $EREF has the form:

^$WINDOW ( wname , gkeyword , gname , ckeyword , ritem , citem , cattribute )

If the error involves a choice attribute for any other type of gadget, $EREF has the form:

^$WINDOW ( wname , gkeyword , gname , ckeyword , item , cattribute )

If the error involves a choice attribute for a menu, $EREF has the form:

^$WINDOW ( wname , mkeyword , mname , ckeyword , item , cattribute )

* 1. Errors

M46 Invalid attribute name.

M47 Invalid attribute value.

M48 Nonexistent window, element or choice.

M49 Invalid attempt to set focus.

M50 Attempt to reference a non-MTERM window in an OPEN command.

M51 Attempt to destroy MTERM window prior to CLOSE.

M52 Required attribute missing.

M53 Invalid argument for font function.

M54 Attempt to create non-modal child of modal parent.

M55 Invalid nested ESTART.

# 

# Conformance

* 1. Implementations
     1. Conforming Implementation

A *conforming implementation* of the MWAPI shall:

a. Correctly implement all features conforming to this standard.

b. Detect all errors required by this standard.

c. Complete the following conformance specification:

XXX version V implements X11.6-YYYY with the following conformance specification:

|  |  |  |
| --- | --- | --- |
| **Implementation Definition****Implementation Definition** | | |
| **Section** | **Feature** | **Option/Value** |
| 4.1.15 | PLATFORM dattribute |  |
|  | Minimum hardware configuration |  |
|  | Operating system name |  |
|  | Operating system version |  |
|  | Required software |  |
|  | Special configuration requirements |  |
|  | Devices emulated by MTERM |  |
|  | Additional controlmnemonics supported for MTERM |  |

d. Use the implementationattribute and implementationvalue name spaces for implementation-specific features.

e. Define all implementation-specific features in the following table.

|  |  |  |
| --- | --- | --- |
| **Implementation-Specific Features****Implementation-Specific Features** | | |
| **Keyword** | **Window/Element/**  **Event/Attribute** | **Description** |
|  |  |  |
|  |  |  |
|  |  |  |

f. Provide, at a minimum, the limits defined in the following table.

| **Portability Limits****Portability Limits** | | | |
| --- | --- | --- | --- |
| **Section** | **Feature** | **Description** | **Limit** |
| 3.3.5  6.5.4 | application-  attribute, AID cattribute | Aggregate size of applicationattribute and AID cattribute values for a process, calculated according to the M local variable storage specification. | M local variable storage limit |
| 5 | wname | Number of characters in window name | 31 |
| 5 |  | Number of windows | 31 |
| 6 |  | Number of elements per M process | 2,047 |
| 6.1 | gname | Number of characters in gadget name | 31 |
| 6.1 |  | Number of gadgets per window | 255 |
| 6.2 | mname | Number of characters in menu name | 31 |
| 6.3 | tname | Number of characters in timer name | 31 |
| 6.3 |  | Number of timers per window | 7 |
| 6.4.8 | CHARMAX elattribute | Maximum number of characters in the text value of a list entry box or text gadget | M string length |
| 6.4.9 | item | Total number of menu choices per window | 255 |
| 6.4.9 | CHOICE elattribute | Number of characters in a single choice | 255 |
| 6.4.9 | item | Number of choices in a list box, list entry box, or long list box. | 1023 |
| 6.4.9 | item | Number of choices in a list button | 31 |
| 6.4.9 | CHOICE elattribute | Aggregate number of characters in all choices for a gadget | 16,383 |
| 6.4.52 | TITLE elattribute | Maximum number of characters in an element title | 255 |
| 6.4.57 | longchars | Maximum number of characters for the text value of a document gadget | 32,767 |
| 6.5.9 | SUBMENU cattribute | Menu depth | 7 |
| 7.1 | drawnum | Aggregate number of drawcommands for a process | 2,047 |

g. Define all features that are not implemented in the following table.

|  |  |  |
| --- | --- | --- |
| **Unimplemented Features****Unimplemented Features** | | |
| **Section** | **Feature** | **Description** |
|  |  |  |
|  |  |  |
|  |  |  |

A conforming implementation may, but is not required to, permit an application to modify the values of attributes for which this standard does not specify Modify access.

* + 1. MDC Conforming Implementation

An *MDC conforming implementation* shall be a conforming implementation except that the conforming document shall be this Standard together with any such current MDC documents that the vendor chooses to implement. The unimplemented features table must be empty in order to have a conforming implementation. The conformance statement shall be in the form:

XXX version V implements X11.6-YYYY with the following conformance specification:

:

:

as modified by the following MDC documents ddd (MDC status sss)

* + 1. MDC Strictly Conforming Implementation

An *MDC strictly conforming implementation* is an MDC conforming implementation that implements only MDC documents with MDC Type A status. An MDC strictly conforming implementation may not permit an application to modify the values of attributes for which this standard does not specify Modify access.

* + 1. MDC Partial Implementation

An *MDC partial implementation* is an implementation that:

a. Provides a subset of the functionality required by this Standard.

b. Conforms in all other respects to this Standard.

c. Documents all areas of non-compliance with this Standard in the unimplemented features table.

* + 1. Multiple Levels of Conformance

An implementation may claim more than one of the previously defined levels of conformance if it provides a mechanism by which the user is able to select a conformance level.

**Index**

*To Be Added*