The M Windowing API: Windows, Ho!

by Gardner S. Trask III

Welcome to the third and final installment of an introduction to the M Windowing Application Program Interface (MWAPI). The first article in the series discussed the paradigm shifts that (D)roll-and-scroll programmers must employ to port to windows.[1] The second article discussed the language syntax and structures of the MDC Type-A MWAPI itself.[2] This article wraps it all together by creating step-by-step a real windows application. Since there is not a lot of syntax and structure background here, please familiarize yourself with the windowing API discussed in previous articles or through some other means.

The Calculator

Instead of creating a huge application, we'll demonstrate a majority of the MWAPI features by developing a relatively simple application—a calculator. This will allow a more robust discussion of the MWAPI features without delving into a lot of non-API application code. This application, when done, will give you a tool that should look similar to figure 1.

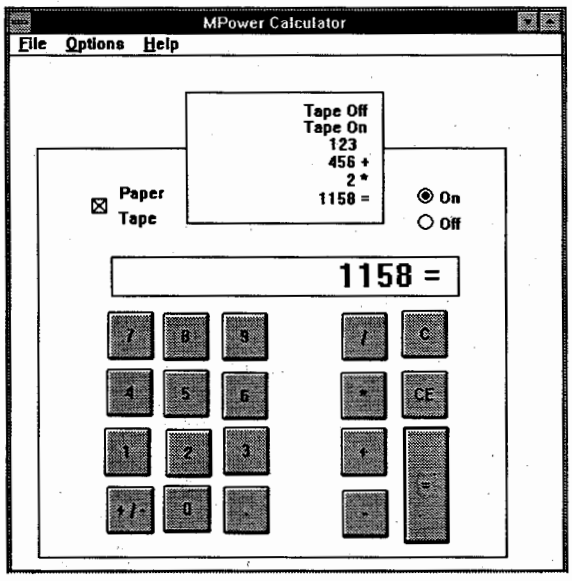

Figure 1. Complete MPower calculator.

And while it is described as a "simple calculator," the reader should note that this application will invoke the following MWAPI elements:

- Menu bars with horizontal and vertical menus;
- Push-button gadgets;
- Radio-button gadgets;
- A text-box gadget;
- A list-box gadget;
- A check-box gadget;
- A symbol gadget;
- A frame gadget;
- Label gadgets;
- Error handling; and
- Multiple windows.

The idea is to create a calculator application, which, when invoked, will allow the client to use a mouse device to point and click the buttons of a calculator and perform the four basic math functions (+ -/*). Users can turn the calculator on or off using a radio button. They can turn on a scrolling-list box to emulate a paper tape via a check box. They can navigate up and down the "paper tape" via a scroll bar, and they can choose various options from a menu such as decimal, binary, or octal calculations.

Calling this "a simple calculator application" may be a bit of an understatement.

Bare Bones

Let us start slowly and build one piece at a time. We will begin by getting the calculator to look like figure 2. This is a calculator that performs the basic arithmetic functions and prints the running values in a text box.

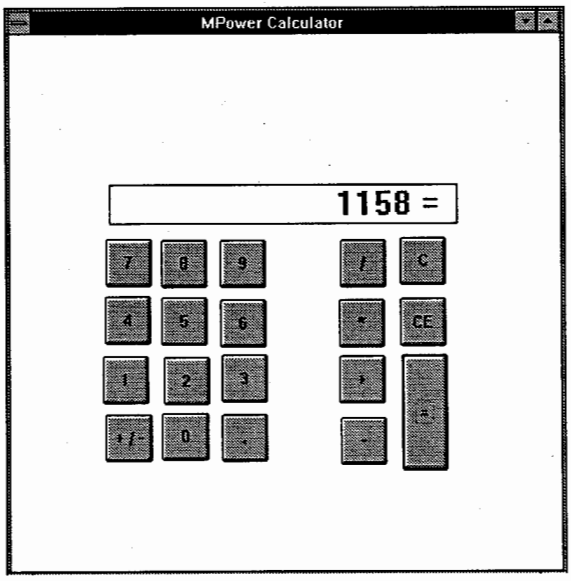

Figure 2. Beginner MPower calculator.

The first thing we should consider is the window. The window represents the "playing field" where our calculator resides. In this case, and for ease of discussion and purposes of brevity, I will use the Microsoft 3.x Windows platform. The user should remember, however, that one of the great advantages of the MWAPI is the intentional MDC design for platform independence. This means that this code should run, *without modification*, on MWAPI implementations of machines that are X Window-based, Macintosh-based, WindowsNT-based, or any other "windows" platform.

The second MWAPI article in this series described the many attributes and options for each window, display, and gadget type. While there may be dozens of options for each of these elements, the programmer can intelligently use the default values of the host windowing system and thus minimize coding.

(At this point we will begin discussion of setting values into globals by hand, although implementors and tools vendors are now, or will soon be, developing graphical-user interface toolkits to allow "point and click" building of the interface. The calculator uses a pre-release version of the MSM-VIEW Builder.)

By using the default values of the host windowing system, only three elements are needed to create my window. I must give it a starting position on my display, a window size value, and a title. I will be setting these values in a personal global and then later MERGEing them into ^$WINDOW. (As you recall from the second article, the effect of the MERGE command with the ^$WINDOW ssvn causes a side effect that creates whatever window or gadget is merged.)

I will set the following window element attributes (the complete global is included at the end of the article):

```
^MPower("MPower","POS")="137,0,PIXEL"
                ,"SIZE")="468,431,PIXEL"
                ,"TITLE")="MPower Calculator"
                ,"TYPE")="APPLICATION"
```

- "MPower" as a first subscript level represents a name for this window, one which can be passed to ^$WINDOW and used in the event of parent/child relationships.
- "POS" describes the starting x,y coordinate on the display and the units of measurement.
- "SIZE" describes the x,y sizing of the window and the units of measurement.
- "TITLE" and "TYPE" should be obvious and are outlined in other documents.

Next are the push buttons that make up the bulk of the interface. Again, by using system defaults I forego having to detail such things as color and active settings, but if clients change their host window-system colors, these preferences get passed right on to my application.

There are two sets of push buttons on my calculator: a numeric keypad and an operator keypad. The only real differences between them are the titles, position, and event callbacks. In a moment we will discuss event processing, but for now we find the only global attributes required for a useful button are a start position, the size, a title, the type of gadget it is, and an event call. For example, the button for the number 7 would look like this in the global:

```
^MPower("MPower","G","BUTTON7","EVENT" ...
... ,"SELECT")=GSTNum^MPower
    ,"POS")="82,220,PIXEL"
    ,"SIZE")="39,41,PIXEL"
    ,"TITLE")="7"
    ,"TYPE")="BUTTON"
```

The numeric buttons (0-9, "." and "+/-") will all have event calls to GSTNum^MPower. The operation keys (*,-,+,/ = C, and CE) will all call GSTOper.

For our simplified calculator we need a repository for the numbers and results as we go along. Here there are several possible options, but the most obvious choice is a text box. The position and size attributes are necessary, but I increased the font size and used the System font for looks. The global looks like this:

```
^MPower("MPower","G","TEXT1" ...
... ,"FFACE")="SYSTEM"
    ,"FSIZE")="20"
    ,"POS")="85,173,PIXEL"
    ,"SIZE")="296,34,PIXEL"
    ,"TITLE")=""
    ,"TYPE")="TEXT"
```

Given the settings of these three objects as well as the remaining numeric and operator keys (with appropriate position, title, and event changes), a `MERGE^$WINDOW("MPower")=^MPower("MPower")` command should yield a window just like figure 2.

The Big Event

As described earlier, certain objects can have assigned events. What this means is that an event handler is, conceptually, a background process that is constantly running. It is started when a routine invokes the ESTART command and sits in the background, checking constantly to see if some event has happened such as a key click, a mouse movement, a mouse-button click, etc. Depending on the object (window, display, or gadget), an event callback can happen on numerous events, such as when a gadget gets or loses focus, or when a selection is made, or when a function key is pressed.

Remember that one of the powerful features of the MWAPI is the automatic inheritance of the base platform's default events. For example, there's no need to write code to resize the window if the host platform supports drag and drop. You have to concern yourself only with events you want to process. In the case of the calculator, pressing a number key calls code to get the value currently in the text box and pad on the pressed key. (Obviously there is more detail to the routine logic; a copy of the routine is included at the end of the article, and is available for download.) Pressing the operator keys makes a different event call to code that either clears the current display entry, clears the running memory, or performs the appropriate calculations based on a memory value and the display value. There could be a separate event callback for each button, but I chose to have all button-click events in one of two callbacks. We will examine the code momentarily.

When the user no longer wants the event handler to run for a window, an ESTOP command turns off the background process.

This is a dramatic oversimplification, but in reality a couple of pages of code and a couple of pages of globals create a fully functioning calculator with the look and feel of the host windowing platform. This code, as written, will run exactly the same under Micronetics Standard MUMPS, DataTree MUMPS, and Digital Equipment Corporation's windows versions with *no required modifications*. More important than vendor portability is platform portability. This code also runs under WindowsNT, Microsoft Windows 3.1 and X-terminal configurations without modification. **This feature is unique to the M language.**

Adding Extras

While the bare-bones calculator does work, we can, with a few additions to the global really punch up the product.

Adding the radio button provides an on/off switch to control the calculator. The global reference looks like this:

```
^MPower("MPower","G","RADIO1" ...
... ,"CHOICE")
    ,"CHOICE",1)                 On
    ,"CHOICE",1,"ACTIVE")        1
    ,"CHOICE",2)                 Off
    ,"CHOICE",2,"ACTIVE")        1
    ,"EVENT","DESELECT")         GSTOff^MPower
    ,"EVENT","SELECT")           GSTOff^MPower
    ,"POS")                      335,106,PIXEL
    ,"SIZE")                     65,48,PIXEL
    ,"TYPE")                     RADIO
    ,"VALUE")                    1
```

To add a running "paper tape" to the calculator, I used a list box. As with many of the gadgets you may use in your application, often you must choose among several possibilities. I like a list box because it automatically adds a scroll bar as the tape gets longer. Users can then scroll up and down through the tape "history" if they desire (with no extra coding on my part). The global reference to add a list box is as follows:

```
^MPower("MPower","G","LIST1","POS")   149,31,PIXEL
                            ,"SIZE")  168,123,PIXEL
                            ,"TITLE")
                            ,"TYPE")  LIST
```

This global puts only the list box on the calculator; filling the list box with values is the responsibility of the routine.

I also wanted to give the user the ability to turn the tape box on and off, so I added a check box. The title for a check box is, by structure, located above the check box; I wanted the title on the right-hand side. Thus, I made the check-box title null, and added two labels to satisfy my requirements. This is the global representation of this check box and label:

```
^MPower("MPower","G","CHECK1" ...
... ,"EVENT","DESELECT")tapebox^MPower
    ,"EVENT","SELECT")   tapebox^MPower
    ,"POS")              69,119,PIXEL
    ,"SIZE")             20,22,PIXEL
    ,"TITLE")
    ,"TYPE")             CHECK
^MPower("MPower","G","LABEL1","POS")    92,110,PIXEL
                            ,"SIZE")    38,16,PIXEL
                            ,"TITLE")   Paper
                            ,"TYPE")    LABEL
```

```
^MPower("MPower","G","LABEL2","POS")   92,132,PIXEL
                               ,"SIZE")  32,18,PIXEL
                               ,"TITLE") Tape
                               ,"TYPE")  LABEL
```

Finally, I decided to frame the calculator. This makes the calculator attractive, but frames are often used to group like objects together. Another place for a frame might have been around the on/off radio buttons. The global setting is:

```
^MPower("MPower","G","zFRAME1","POS") 23,71,PIXEL
                               ,"SIZE")398,360,PIXEL
                               ,"TITLE")
                               ,"TYPE")FRAME
```

(Due to a gap in the current standard, the order of objects is undefinable. By calling the object "zFRAME1", I can utilize a vendor's "ordering" to get my frame to load properly. Otherwise, some objects may have been hidden by the frame. This gap has been recognized, and a solution is being recommended for the next version of the MWAPI standard.)

To be even fancier, I have added a call in the routine to pop up a special window if the user tries to divide by zero. This window contains a "warning" symbol, a "warning" label, and an "O.K." button. This window could have been a child of the main menu, but personal style did not dictate it so. The global for this window can be found in the attached global under `^MPower("zero")` (see figure 3).

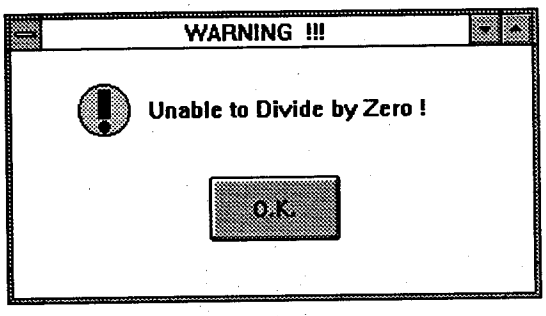

Figure 3. The warning window.

I also have added a menu bar to the final version. It contains horizontal options for File, Options, and Help. It also contains vertical options under each horizontal choice. Under File is the Quit option, under Help is the About MPower Calculator option, and under Options are binary, decimal, and octal choices. The idea is that, in a future version of MPower, these choices would quit, pull up an "About" window, or (in the case of Options) set a flag to calculate in decimal, binary, or octal mode and to turn nonessential numeric buttons to grey by setting the buttons to inactive.

The Code

Here is what the routine MPower is doing. At the calling point `D ^MPower`, the system kills any leftover calculator windows, sets the initialization variable, `MERGE`s the global to the `^$WINDOW` global, and starts the event handler.

As events come, calls to different tags perform different functions:

`GSTNum`	Calls to this tag get the current text-box value and pad on the key that was last clicked, checking for existing periods, "+/-" signs and leading zeros.
`GSTOper`	This event call either clears the display (CE), clears the tape and display and memory (C), or performs the appropriate arithmetic function using the memory amount and display amount.
`GSTOff`	This calls for the calculator to be turned off, killing the window, and stopping the event handler.
`tapebox`	Calls an event to turn the "paper tape" list box on or off.
`zdivide`	Calls the "divide by zero" warning window.

The Final Word

This example of the calculator has been designed to give the reader an overview of the MWAPI. More detailed documentation is available, and the user is encouraged to seek out the appropriate periodicals and standards. The complete global and routine is available for download (free of charge) through the NEMUG Bulletin Board. Dial in at 508-921-6681 (8-N-1) and look under the Files submenu for Mpower.ROU and Mpower.GBL.

Gardner S. Trask III is senior programmer/analyst for Matthew Thornton Health Plan in Nashua, New Hampshire, and owns MPower Computer Consulting, which holds the MPower trademark. This article does not necessarily reflect the views of his employers. He may be reached at 508-927-7637, via the NEMUG BBS at 508-921-6681, or through Internet at trask@world.std.com.

Endnotes

1. G.S. Trask III, "The M Windowing API: Expansive Tool or Expensive Toy?" *M Computing*, 2:1 (February 1994), 45–48.
2. MDC refers to the MUMPS Development Committee. G.S. Trask III, "The M Windowing API: The Tools," *M Computing*, 2:2 (April 1994), 47–52.

Appendix. The MPower global and routine to create the sample calculator through the M Windowing Application Program Interface.

```
MPower    ; MPower Calculator code - Gardner S. Trask III [ 07/13/93 11:15 PM ]
          ; Copyright 1994 - MPower Computer Consultants - All rights reserved
Al        ;
          k ^$W("MPower")
          s $zt="ERRExit"
          s newflag=1
          M ^$W("MPower")=^MPower("MPower")
          d clrtape,tapebox,clear
          ESTART
          ;
GSTExit   ; GSTExit
          ;general call for ESTOP
          ESTOP
          Q
ERRExit   ESTOP
          Q
          ;
GSTOff    k ^$W("MPower")
          g GSTExit
          -
```

```
GSTNum    ; GSTNum
          ;add number to display
          d getdisp
          d getkey
          ;
          i key="." d
          .    Q:dval["."
          .    s dval=dval_"."
          .    d putdisp
          ;
          i key="+ / -" d
          .    s dsign=$S(dsign="-":"+",1:"-")
          .    d putdisp
          ;
          i key>0 d
          .    i newflag s dval=key,dsign="" d putdisp q
          .    i +dval>0!(dval[".") s dval=dval_key
          .    e s dval=key
          .    d putdisp
          ;
          i key="0" d ;order of logic is important here
          .    i dval["." s dval=dval_0 d putdisp q
          .    i newflag s dval=0,dsign="" d putdisp q
          .    i dval=0 q
          .    s dval=dval_0
          .    d putdisp
          ;
          s newflag=0
          Q
          ;
GSTOper   ;process for operator key-clicks
          d getkey
          i key="C" d clear q
          i key="CE" d cleare q
          i key="=" d equal q
          d oper
          q
          ;
getkey    ;get keyclick title value
          s key=^$EVENT("ELEMENT")
          S key=$P(key,",",2)
          s key=^$W("MPower","G",key,"TITLE")
          q
          -
clear          ;clear key
          s dsign=" ",doper="C",dval=0,newflag=1
          s msign=" ",moper="C",mval=0
          i tapeon s tapeval="0 C" d puttape
          d putdisp
          q
          ;
cleare         ;clear entry
          s dsign=msign,doper=moper,dval=mval d putdisp
          s newflag=1
          q
oper           ;operator manipulation
          s nmoper=$s(moper="C":"+",moper="=":"+",1:moper) d calc i $G(zdflag) q
          i tapeon s tapeval=dsign_dval_" "_$S(moper="C":"",moper="=":"",1:moper) d puttape
          s (mval,dval)=$e(rval,1,15),(msign,dsign)=rsign,(moper,doper)=key
          s newflag=1
          d putdisp
          q
          -
```

```
equal           q:newflag
          s nmoper=moper d calc i $G(zdflag) q
          i tapeon b d
          .    s tapeval=dsign_dval_" "_moper d puttape
          .    s tapeval=rsign_rval_" "_key d puttape
          .    s tapeval="" d puttape
          s mval=$e(rval,1,15),msign=rsign,moper=key
          s dval=rval,dsign=rsign,doper=key d putdisp
          s newflag=1
          q
calc            ;calculate new values
          d getdisp s zdflag=0
          s mem=$S(msign=" ":"",1:msign)_mval
          s disp=$s(dsign=" ":"",1:dsign)_dval
          i moper="/",(disp=0) d zdivide s zdflag=1 q
          s result=msign_mval
          i nmoper="+" s result=mem+disp
          i nmoper="-" s result=mem-disp
          i nmoper="*" s result=mem*disp
          i nmoper="/" s result=mem/disp
          i moper="=" s result=disp
          s rsign=$S(result<0:"-",1:"")
          s rval=$S(result>0:+result,1:result*-1)
          q
          ;
getdisp   ;get value in display
          s old=^$W("MPower","G","TEXT1","VALUE")
          s y=$L(old) f I=1:1:y q:$E(old)'=" " s old=$E(old,2,y) ;strip leading spaces
          s dsign=$E(old) i dsign'="+",(dsign'="-") s dsign=" ",old=" "_old
          s old=$E(old,2,$L(old))
          s doper=$E(old,$L(old))
          s dval=$E(old,1,($L(old)-2))
          q
putdisp   ;put value to display
          i newflag,(dval'=0),(doper="C") s doper=""
          i newflag,(doper="=") s doper=""
          s xval=dsign_$e(dval,1,15)_" "_$S($L(doper):doper,1:" ")
          S ^$W("MPower","G","TEXT1","VALUE")=$j(xval,20)
          q
tapebox   s tapeon=+$G(^$W("MPower","G","CHECK1","VALUE"))
          s tapeval="Tape "_$S(tapeon:"On",1:"Off") d puttape
          q
clrtape   k ^$W("MPower","G","LIST1","CHOICE")
          q
puttape    b  n x
          s x=$O(^$W("MPower","G","LIST1","CHOICE",""),-1)
          s x=x+1
          s ^$W("MPower","G","LIST1","CHOICE",x)=$J(tapeval,25)
          s ^$W("MPower","G","LIST1","TOPSHOW")=x
          ;s ^$W("MPower","G","LIST1","TOPPOS")=x
          q
          -
zdivide    M ^$W("zero")=^MPower("zero")
          q
zdbut          k ^$W("zero")
          d cleare
          q
          -
notyet    ;callback for unimplemented features
          q
```

```
11:22 PM 13-JUL-93
^MPower("MPower","G","BUTTON1","EVENT","SELECT")  GSTNum^MPower
^MPower("MPower","G","BUTTON1","POS")             79,320,PIXEL
^MPower("MPower","G","BUTTON1","SIZE")            39,41,PIXEL
^MPower("MPower","G","BUTTON1","TITLE")           1
^MPower("MPower","G","BUTTON1","TYPE")            BUTTON
^MPower("MPower","G","BUTTON10","EVENT","SELECT") GSTNum^MPower
^MPower("MPower","G","BUTTON10","POS")            129,369,PIXEL
^MPower("MPower","G","BUTTON10","SIZE")           40,41,PIXEL
^MPower("MPower","G","BUTTON10","TITLE")               0
^MPower("MPower","G","BUTTON10","TYPE")           BUTTON
^MPower("MPower","G","BUTTON11","EVENT","SELECT") GSTNum^MPower
^MPower("MPower","G","BUTTON11","POS")            180,370,PIXEL
^MPower("MPower","G","BUTTON11","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON11","TFSIZE")         25
^MPower("MPower","G","BUTTON11","TITLE")               .
^MPower("MPower","G","BUTTON11","TYPE")           BUTTON
^MPower("MPower","G","BUTTON12","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON12","POS")            281,221,PIXEL
^MPower("MPower","G","BUTTON12","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON12","TITLE")               /
^MPower("MPower","G","BUTTON12","TYPE")           BUTTON
^MPower("MPower","G","BUTTON13","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON13","POS")            280,272,PIXEL
^MPower("MPower","G","BUTTON13","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON13","TITLE")               *
^MPower("MPower","G","BUTTON13","TYPE")           BUTTON
^MPower("MPower","G","BUTTON14","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON14","POS")            280,320,PIXEL
^MPower("MPower","G","BUTTON14","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON14","TITLE")               +
^MPower("MPower","G","BUTTON14","TYPE")           BUTTON
^MPower("MPower","G","BUTTON15","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON15","POS")            281,373,PIXEL
^MPower("MPower","G","BUTTON15","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON15","TFSIZE")         25
^MPower("MPower","G","BUTTON15","TITLE")               -
^MPower("MPower","G","BUTTON15","TYPE")           BUTTON
^MPower("MPower","G","BUTTON16","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON16","POS")            332,219,PIXEL
^MPower("MPower","G","BUTTON16","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON16","TITLE")               C
^MPower("MPower","G","BUTTON16","TYPE")           BUTTON
^MPower("MPower","G","BUTTON17","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON17","POS")            332,271,PIXEL
^MPower("MPower","G","BUTTON17","SIZE")           39,41,PIXEL
^MPower("MPower","G","BUTTON17","TITLE")               CE
^MPower("MPower","G","BUTTON17","TYPE")           BUTTON
^MPower("MPower","G","BUTTON18","EVENT","SELECT") GSTOper^MPower
^MPower("MPower","G","BUTTON18","POS")            333,319,PIXEL
^MPower("MPower","G","BUTTON18","SIZE")           39,100,PIXEL
^MPower("MPower","G","BUTTON18","TFSIZE")         25
^MPower("MPower","G","BUTTON18","TITLE")               =
^MPower("MPower","G","BUTTON18","TYPE")           BUTTON
^MPower("MPower","G","BUTTON19","EVENT","SELECT") GSTNum^MPower
^MPower("MPower","G","BUTTON19","POS")            81,370,PIXEL
^MPower("MPower","G","BUTTON19","SIZE")           40,41,PIXEL
^MPower("MPower","G","BUTTON19","TITLE")               + / -
^MPower("MPower","G","BUTTON19","TYPE")           BUTTON
^MPower("MPower","G","BUTTON2","EVENT","SELECT")  GSTNum^MPower
^MPower("MPower","G","BUTTON2","POS")             131,321,PIXEL
^MPower("MPower","G","BUTTON2","SIZE")            39,41,PIXEL
^MPower("MPower","G","BUTTON2","TITLE")           2
^MPower("MPower","G","BUTTON2","TYPE")            BUTTON
^MPower("MPower","G","BUTTON3","EVENT","SELECT")  GSTNum^MPower
^MPower("MPower","G","BUTTON3","POS")             180,319,PIXEL
^MPower("MPower","G","BUTTON3","SIZE")            39,41,PIXEL
^MPower("MPower","G","BUTTON3","TITLE")           3
^MPower("MPower","G","BUTTON3","TYPE")            BUTTON
^MPower("MPower","G","BUTTON4","EVENT","SELECT")  GSTNum^MPower
^MPower("MPower","G","BUTTON4","POS")             81,269,PIXEL
^MPower("MPower","G","BUTTON4","SIZE")            39,41,PIXEL
```

```
^MPower("MPower","G","BUTTON4","TITLE")            4
^MPower("MPower","G","BUTTON4","TYPE")             BUTTON
^MPower("MPower","G","BUTTON5","EVENT","SELECT")   GSTNum^MPower
^MPower("MPower","G","BUTTON5","POS")              129,270,PIXEL
^MPower("MPower","G","BUTTON5","SIZE")             39,41,PIXEL
^MPower("MPower","G","BUTTON5","TITLE")            5
^MPower("MPower","G","BUTTON5","TYPE")             BUTTON
^MPower("MPower","G","BUTTON6","EVENT","SELECT")   GSTNum^MPower
^MPower("MPower","G","BUTTON6","POS")              179,272,PIXEL
^MPower("MPower","G","BUTTON6","SIZE")             39,41,PIXEL
^MPower("MPower","G","BUTTON6","TITLE")            6
^MPower("MPower","G","BUTTON6","TYPE")             BUTTON
^MPower("MPower","G","BUTTON7","EVENT","SELECT")   GSTNum^MPower
^MPower("MPower","G","BUTTON7","POS")              82,220,PIXEL
^MPower("MPower","G","BUTTON7","SIZE")             39,41,PIXEL
^MPower("MPower","G","BUTTON7","TITLE")            7
^MPower("MPower","G","BUTTON7","TYPE")             BUTTON
^MPower("MPower","G","BUTTON8","EVENT","SELECT")   GSTNum^MPower
^MPower("MPower","G","BUTTON8","POS")              129,221,PIXEL
^MPower("MPower","G","BUTTON8","SIZE")             39,41,PIXEL
^MPower("MPower","G","BUTTON8","TITLE")            8
^MPower("MPower","G","BUTTON8","TYPE")             BUTTON
^MPower("MPower","G","BUTTON9","EVENT","SELECT")   GSTNum^MPower
^MPower("MPower","G","BUTTON9","POS")              179,221,PIXEL
^MPower("MPower","G","BUTTON9","SIZE")             39,41,PIXEL
^MPower("MPower","G","BUTTON9","TITLE")            9
^MPower("MPower","G","BUTTON9","TYPE")             BUTTON
^MPower("MPower","G","CHECK1","EVENT","DESELECT") tapebox^MPower
^MPower("MPower","G","CHECK1","EVENT","SELECT")   tapebox^MPower
^MPower("MPower","G","CHECK1","POS")               69,119,PIXEL
^MPower("MPower","G","CHECK1","SIZE")              20,22,PIXEL
^MPower("MPower","G","CHECK1","TITLE")
^MPower("MPower","G","CHECK1","TYPE")              CHECK
^MPower("MPower","G","LABEL1","FCOLOR")            30,30,30
^MPower("MPower","G","LABEL1","POS")               92,110,PIXEL
^MPower("MPower","G","LABEL1","SIZE")              38,16,PIXEL
^MPower("MPower","G","LABEL1","TITLE")             Paper
^MPower("MPower","G","LABEL1","TYPE")              LABEL
^MPower("MPower","G","LABEL2","POS")               92,132,PIXEL
^MPower("MPower","G","LABEL2","SIZE")              32,18,PIXEL
^MPower("MPower","G","LABEL2","TITLE")             Tape
^MPower("MPower","G","LABEL2","TYPE")              LABEL
^MPower("MPower","G","LIST1","CHOICE")
^MPower("MPower","G","LIST1","POS")                149,31,PIXEL
^MPower("MPower","G","LIST1","SIZE")               168,123,PIXEL
^MPower("MPower","G","LIST1","TITLE")
^MPower("MPower","G","LIST1","TYPE")               LIST
^MPower("MPower","G","RADIO1","CHOICE")
^MPower("MPower","G","RADIO1","CHOICE",64)         On
^MPower("MPower","G","RADIO1","CHOICE",64,"ACTIVE")    1
^MPower("MPower","G","RADIO1","CHOICE",96)         Off
^MPower("MPower","G","RADIO1","CHOICE",96,"ACTIVE")    1
^MPower("MPower","G","RADIO1","EVENT","DESELECT") GSTOff^MPower
^MPower("MPower","G","RADIO1","EVENT","SELECT")  GSTOff^MPower
^MPower("MPower","G","RADIO1","FCOLOR")            0,0,0
^MPower("MPower","G","RADIO1","POS")               335,106,PIXEL
^MPower("MPower","G","RADIO1","SIZE")              65,48,PIXEL
^MPower("MPower","G","RADIO1","TITLE")
^MPower("MPower","G","RADIO1","TYPE")              RADIO
^MPower("MPower","G","RADIO1","VALUE")             64
^MPower("MPower","G","TEXT1",FFACE")               SYSTEM
^MPower("MPower","G","TEXT1","FSIZE")              20
^MPower("MPower","G","TEXT1","POS")                85,173,PIXEL
^MPower("MPower","G","TEXT1","SIZE")               296,34,PIXEL
^MPower("MPower","G","TEXT1","TITLE")
^MPower("MPower","G","TEXT1","TYPE")               TEXT
^MPower("MPower","G","TEXT1","UNITS")              CHAR
^MPower("MPower","G","zFRAME1","POS")              23,71,PIXEL
^MPower("MPower","G","zFRAME1","SIZE")             398,360,PIXEL
^MPower("MPower","G","zFRAME1","TITLE")
^MPower("MPower","G","zFRAME1","TYPE")             FRAME
^MPower("MPower","M","File","CHOICE",1)            &Quit
```

```
^MPower("MPower","M","File","CHOICE",1,"ACTIVE")  1
^MPower("MPower","M","File","CHOICE",1,"EVENT","SELECT")notyet^MPower
^MPower("MPower","M","File","CHOICE",1,"EVENT","SELECT","ENABLE")     1
^MPower("MPower","M","File","ID")                  17
^MPower("MPower","M","File","UNITS")               PIXEL
^MPower("MPower","M","File","VISIBLE")
^MPower("MPower","M","Help","CHOICE",1)            &About MPower Calculator
^MPower("MPower","M","Help","CHOICE",1,"ACTIVE")  1
^MPower("MPower","M","Help","CHOICE",1,"EVENT","SELECT")    notyet^MPower
^MPower("MPower","M","Help","CHOICE",1,"EVENT","SELECT","ENABLE")     1
^MPower("MPower","M","Help","ID")                  18
^MPower("MPower","M","Help","UNITS")               PIXEL
^MPower("MPower","M","Help","VISIBLE")
^MPower("MPower","M","Options","CHOICE",1)         &Binary
^MPower("MPower","M","Options","CHOICE",1,"ACTIVE")    1
^MPower("MPower","M","Options","CHOICE",1,"EVENT","SELECT") notyet^MPower
^MPower("MPower","M","Options","CHOICE",1,"EVENT","SELECT","ENABLE")  1
^MPower("MPower","M","Options","CHOICE",2)         &Decimal
^MPower("MPower","M","Options","CHOICE",2,"ACTIVE")    1
^MPower("MPower","M","Options","CHOICE",2,"EVENT","SELECT") notyet^MPower
^MPower("MPower","M","Options","CHOICE",2,"EVENT","SELECT","ENABLE")  1
^MPower("MPower","M","Options","CHOICE",3)         &Octal
^MPower("MPower","M","Options","CHOICE",3,"ACTIVE")    1
^MPower("MPower","M","Options","CHOICE",3,"EVENT","SELECT") notyet^MPower
^MPower("MPower","M","Options","CHOICE",3,"EVENT","SELECT","ENABLE")  1
^MPower("MPower","M","Options","ID")
^MPower("MPower","M","Options","UNITS")            PIXEL
^MPower("MPower","M","Options","VISIBLE")
^MPower("MPower","M","MAIN","CHOICE",1)            &File
^MPower("MPower","M","MAIN","CHOICE",1,"ACTIVE")  1
^MPower("MPower","M","MAIN","CHOICE",1,"SUBMENU") File
^MPower("MPower","M","MAIN","CHOICE",2)            &Options
^MPower("MPower","M","MAIN","CHOICE",2,"ACTIVE")  1
^MPower("MPower","M","MAIN","CHOICE",2,"SUBMENU") Options
^MPower("MPower","M","MAIN","CHOICE",3)            &Help
^MPower("MPower","M","MAIN","CHOICE",3,"ACTIVE")  1
^MPower("MPower","M","MAIN","CHOICE",3,"SUBMENU") Help
^MPower("MPower","M","MAIN","ID")
^MPower("MPower","M","MAIN","UNITS")               PIXEL
^MPower("MPower","M","MAIN","VISIBLE")
^MPower("MPower","MENUBAR")                        MAIN
^MPower("MPower","POS")                            137,0,PIXEL
^MPower("MPower","SIZE")                           468,431,PIXEL
^MPower("MPower","SIZEMIN")
^MPower("MPower","SIZEWIN")                        476,477,PIXEL
^MPower("MPower","TITLE")                          MPower Calculator
^MPower("MPower","TYPE")                           APPLICATION
^MPower("zero","G","BUTTON 1","EVENT","SELECT")    zdbut^MPower
^MPower("zero","G","BUTTON 1","POS")               121,79,PIXEL
^MPower("zero","G","BUTTON 1","SIZE")              80,40,PIXEL
^MPower("zero","G","BUTTON 1","TITLE")             O.K.
^MPower("zero","G","BUTTON 1","TYPE")              BUTTON
^MPower("zero","G","LABEL 1","POS")                84,29,PIXEL
^MPower("zero","G","LABEL 1","SIZE")               170,20,PIXEL
^MPower("zero","G","LABEL 1","TFSIZE")             10
^MPower("zero","G","LABEL 1","TITLE")              Unable to Divide by Zero!
^MPower("zero","G","LABEL 1","TYPE")               LABEL
^MPower("zero","G","SYMBOL 1","POS")               41,21,PIXEL
^MPower("zero","G","SYMBOL 1","RESOURCE")          M,WARN
^MPower("zero","G","SYMBOL 1","TYPE")              SYMBOL
^MPower("zero","POS")                              183,166,PIXEL
^MPower("zero","SIZE")                             319,153,PIXEL
^MPower("zero","SIZEMIN")
^MPower("zero","SIZEWIN")                          327,180,PIXEL
^MPower("zero","TITLE")                            WARNING !!!
^MPower("zero","TYPE")                             APPLICATION
```