MGWalk Programmer's Utility for Windows

Version 1.0 - June, 1992

by Mark Gamber

MGWalk (c)1992 by Mark Gamber

Version 1.0

Changes

Changes:

Using standard C library.
Class list.
Bitmap display.
Bitmap copy to clipboard.
Save and Print functions
Dialog box display.

MGWalk Overview:

Windows job is primarily that of a memory manager. Everything running in Windows is comprised of different objects such as **modules**, **tasks**, **data** and so on. All these objects lumped together forms the "**Global Heap**", which this utility dissects and displays.

The program displays object addresses, handles, module and module executable pathname, object size and more. If the object is an icon, cursor or bitmap, it may be viewed in native form. Bitmaps may even be copied to the clipboard! Local heap walks automatically occur if the object has a local heap. If nothing else, hexidecimal and ASCII data may be viewed for nearly object.

Methods are available for displaying currently loaded modules, running tasks and registered window classes. You may check system resources and scan the display for window information.

See also: **Walking the Global Heap**

Walking the Global Heap:

The program walks the global heap on startup, when notified and on demand by the user. You may select from four sort methods, available from the "**Sort**" menu in the "**Options**" menu. Upon selecting a sort method, a global heap walk begins. You may also begin using the current style by selecting "**Walk Heap**" in the "**Info**" menu.

A global heap walk can take several seconds. Because of this, it allows other programs to run in the background. It takes a bit longer, but it's not noticed. Also, for a reason yet unexplained to me, there are occasions the heap cannot be peered into and the program will go into a "state". You notice this when the "Working" dialog box doesn't go away for quite some time. Your best bet at this point is to shut the program down and start it up again. I've been informed it's a problem with the TOOLHELP module, but not what the problem is or how to work around it so I have to grin and bear it too.

See also: **What is the Heap?**, **Heap Objects**

Heap Objects:

Everything in the Global heap is an object, though some are more interesting than others. You may look at the contents of nearly any object on the global stack by double-clicking the listbox item of interest or highlighting the item and selecting "**Show Data**" from the "**Info**" menu. If the block selected is able to be peered into, it is displayed in a dialog box in hexadecimal and ASCII character form. The address is displayed as the global address plus the offset per line for the length of the block. Note that large blocks take a while to scan and display, so prepare to wait if you're looking into big data chunks.

Objects labeled "**Icon**", "**Cursor**", "**Dialog**" and some "**Priv. Data**" items may be viewed in their "native form". That is, an icon is displayed in hex. and ASCII and you can also see the icon itself. Cursors, also may be displayed. Dialog boxes are displayed as a non-working dialog with all the controls available. The controls do not respond, however. Bitmaps often reside as "**Priv. Data**" and are owned by the GDI module. If you display a bitmap, it is displayed in bitmap form if compatible with your display. In addition, you may copy a bitmap to the clipboard by selecting "**Copy**" in the system menu of the small window display.

There is no "Bitmap Flag", so a best guess approach is taken to determine if an object is a displayable bitmap. A defined header exists, so the header is checked and if the values seem reasonable, the module name is checked. If the module is GDI and the size and color values look ok, it is assumed a bitmap. This assumption can be wrong either way. That is, don't expect to see every bitmap in your system since they may not be recognized. In addition, some things may look like bitmaps but really aren't. Try to display them and the program dies.

Private data is data that is owned by a module for that module's use and nothing else's. Therefore, it can be proprietary data and is thus termed Private Data.

BACK ABOUT EXIT

What is the "Heap?"

The "Heap", as it is known, is more or less just a collection of objects handled in an organized fashion. Through Windows memory management, code, data, icons, bitmaps, cursors and all manners of objects reside in the same machine at the same time. Through debugging methods provided, this utility scans the heap, dissecting the data types and displaying information about the objects found.

Windows manages a global heap from which global objects may be created. In addition, an application may manage, through Windows, a "local heap" which works the same way and may be dissected the same way, displaying the local heap in the same basic way.

You can find all sorts of interesting data in the Windows heap. Some objects may be displayed in native form by double-clicking the global object. The objects currently supported are Icons, Cursors, Bitmaps and Dialog boxes.

See also: **Walking the Heap**

Update Notification:

Selecting "**Updates...**" in the "**Options**" menu displays a dialog box with four notification sources. Checked sources are active and unchecked are inactive. What does this mean?

If an checked action occurs, such as a DLL loading or a new task starting, a global heap walk is started and control is given to MGWalk. In a heavily used system, having too many or the "wrong" notifications set can slow the system to nearly a complete halt, so be careful.

The "wrong" notification is a judgement based on the amount of re-walking the program does. If your system isn't doing much but scanning the heap, chances are there's a lot of what you have enabled going on in the system. Generally, you only need "New Tasks" and "Started DLL" enabled, if anything at all.

BACK ABOUT EXIT

Registered Classes:

Windows keeps a list of window classes registered with the system. Classes include the edit box, list box, button and so on, in addition to application spawned classes. Selecting "**Classes**" in the "**Info**" menu displays a listbox with a list of currently registered window classes, class instance and extra byte allocations.

Program Modules:

Programs are made up of modules, each of which has a module handle. You may obtain a current list of loaded modules by selecting "**Modules**" from the "**Info**" menu. Displayed are the module name, handle, executable pathname and lock count of the module.

Running Tasks:

Each application is given an instance handle, a module handle and a task handle. The task handle is used in rare occasions when a message needs to be sent to an application with no window. In this case, the task handle is the key to obtaining stack data and other application default memory usage.

Select "**Tasks**" in the "**Info**" menu to display a list of currently running tasks. You may double-click a task to display more information on the task, if available.

Double-clicking a task displays a dialog box which shows local heap, stack and static data areas. In addition, a real-time, graphic representation is displayed, including the current stack pointer.

BACK ABOUT EXIT

Window Information:

Selecting "**Windows**" in the "**Info**" menu displays a dialog box for presenting window information and hides all the other dialog boxes and the main window for the duration of the Window Info dialog box. Press the "**New**" button to begin scanning the display or "**Close**" to exit, bringing back the main window.

When scanning starts, the cursor changes to a crosshair and any window passed over has it's various attributes displayed in the dialog box. Press the left button on a window of interest to halt scanning, saving the attributes of the window clicked on.

Program Preferences:

You may set the background color of the listbox in the main window by selecting "**Back Color**" in the "**Options**" menu. Colors are set using the standard common dialog box.

Text color and font may be changed by selecting "**Text Style**" in the "**Options**" menu. Font and color selection are made using the common dialog box.

"**Omit**", found, in the "**Options**" menu, prevents certain objects from being displayed and, more importantly, taking up memory. By default, "**Free**" is enabled which prevents "Free" heap objects from being displayed. You may also halt display of Code and Private objects by selecting the menu options.

Changes are saved between sessions.

Registering this Program:

Sorry, but this program is not free. It took a lot of effort to get the program this far. If you find the program to be useful and intend to continue using it, the author requests a fee of **$20** in check or money order form. In return, you will receive a registration number to make the box go away as well as notification of updates. The author may be reached at:

Mark Gamber
18 Village Dr.
Lancaster, Pa   17601

America Online E-Mail: **PCA MarkG**
Internet E-Mail: **pcamarkg@aol.com**

System Resources: