Toolbar

2.11

Juan M. Aguirregabiria

Fisika Teorikoa. Zientzi Fakultatea
Euskal Herriko Unibertsitatea
P.K. 644, 48080 Bilbo (Spain)
Internet: `wtpagagj@lg.ehu.es`

October 1993

Copyright

The program **`TOOLBAR.EXE`**, the help file **`TOOLBAR.HLP`**, the documentation file **`TOOLBAR.WRI`**, and all the remaining files in this package are copyrighted:

Other brand and product names are trademarks or registered trademarks of their respective holders.

Toolbar is FreeWare; there is no charge for using it and it may be distributed freely so long as the files are kept together and unaltered. You may neither sell nor profit from distribution of *Toolbar* in any way.

Disclaimer

In no event will the Author be liable to users for any damages, including but not limited to any lost profits, lost savings or other incidental or consequential damages arising out of the use or the inability to use this program, even if the Author has been advised of the possibility of such damages, or for any claim by other party.

Hardware and software requirements

To run *Toolbar* you need Microsoft Windows 3.1 and the hardware necessary to run it.

For Spanish-speaking users
Para hispanohablantes

Existe una versión castellana del programa y del fichero de ayuda que puede obtenerse directamente del autor.

New in this version

New in version 2.11

Corrected bugs:

Selecting both scroll bars crashed the program.
When all tools were visible an empty space was displayed instead of the scroll bar.
The first time that a toolbar was minimized it could be located outside the visible screen.
Winmine (and maybe other programs) are now correctly activated.
Negative values of **Frequency** are no longer set to 0.
After installing an utility icon, programs dropped from *File Manager* were incorrectly installed.

The directory of the first program in a **`GRP`** file was ignored.

There were some problems with drag-and-drop when using *Norton Desktop* instead of *File Manager*.

New features:

When **Titles** is disabled the active element may optionally appear in a red frame if **Button like** is enabled or **Frame** is different from zero.

To support multiple users on a network, if *Toolbar* is the shell and **`TOOLBAR.INI`** is not found in *Toolbar*'s directory, it will be searched for (and created if not found) in the local Windows directory.

Configuration changes can optionally be saved dynamically (as they happen).

There is a new way to move a window.

The *Trashcan* is now smarter when dealing with logical drives created by **`SUBST`** and analogous commands.

Two new quick exit utility buttons have been added.

It is possible to force the exit from Windows when a toolbar is closed. This is especially useful when *Toolbar* fails to recognize that it is the current shell.

It is possible to install a program in a toolbar by picking one of its windows with the mouse.

It is possible to disable the undocumented drag-and-drop features of Windows 3.1 in order to be able to arrange elements under Windows NT.

New in version 2.00

Corrected bugs:

The entries **`run`** and **`load`** in **`WIN.INI`** were interpreted incorrectly.

Blank command lines in the **Run**... dialog box caused problems in the history list.

An additional window was visible while a DOS program was being initialized.

The Spanish help file was included by mistake in some **`TLBAR111.ZIP`** files.

The warning message to indicate that two tools had the same name did not get focus.

When *Toolbar* was the shell, invoking Windows with an argument did not run the program.

Inter-toolbar copy of elements is completed.

New features:

A single click with the right mouse button is now equivalent to a double-click in the left button and it will start a tool (or insert a new one if performed over a void location). In this way, the usual Windows conventions are retained while impatient users are provided with a faster method.

A single click with the right mouse button while the **`Alt`** key is held down is now equivalent to **`Alt+Enter`**: it will open a **Properties** dialog box.

When a new toolbar is started, the options (**Small font, Titles, Button like, Top window, Frame, Frequency, ...**) of the current toolbar are inherited.

The toolbar title bar can be suppressed. The system menu can still be accessed with **`Alt+Enter`** (to move or minimize the toolbar or to access the remaining entries: **Options**, etc.) or by installing a menu icon, as described below.

It is possible to display only some rows of elements. The visible rows are changed with pg up and pg down or by means of an optional scroll bar. Furthermore, the toolbar may optionally have a border that allows resizing it with the mouse.

If **Frequency** is negative, the actual frequency is its negative value in milliseconds.

Free resources percentage is now the minimum of graphics and other resources. In previous versions only graphics resources were taken into account.

Remembering of icon position is now optional.

Toolbar is now extensible and user provided utilities can be easily added. Sample source code is provided.

Some utilities are now provided: trashcan, clock, menu, memory, resources, info and blank icon.

The hourglass cursor is used while a tool is starting.

New in version 1.11

This version is to correct a nasty bug that gave a **`Not enough memory`** error when trying to add an element to a free toolbar. I apologize for any inconvenience.

New in version 1.10

The history list feature has been added to the **Run...** dialog box.

You can use **Advanced options** in the **Program** command line. This lets you control the way in which dropped files are passed to a tool. You can even prompt the user before starting the tool.

You have more control over the directory in which a tool will start. *Toolbar* will use **Directory** if available, the first dropped file's directory or the program's directory if anything else fails.

A toolbar can now be hidden (instead of minimized or quit) when one of its tools is used.

The **program** argument is supported when invoking Windows - i.e. **program** is automatically started.

Some minor bugs leading to memory leaks have been fixed.

More specific error messages are now issued when a tool cannot be started.

The pointer appearance when changing elements order by dragging now looks much better. In fact, true drag-and-drop is now used.

One can now drag elements onto and **from** *Toolbar*.

Toolbar can now use any *Program Manager*'s group file as **INI** file. It will be automatically translated to *Toolbar*'s format.

Toolbar's caption is now updated only if something has changed. This minimizes the potential flicker of some underlying windows.

Acknowledgments

I am indebted to Matt Frost who corrected the poor English of a previous version of this file. The remaining errors have certainly been introduced by myself when adding the text corresponding to the last improvements in the program.

Julio Pons suggested me some programming tricks that are used in this version of *Toolbar*. He is the author of a program (*Desktop Launcher*) similar to *Toolbar*.

I want also to acknowledge very stimulating feedback from my brother Javi and many other usersespecially those that e-mailed me their suggestions and bug reports: Geoff Bishel, Olaf Winkelhake, Ronnie (Rocker), Josu Oyanguren, Frank Robijn, Michael John Raine, Tian (DARSTELLUNG), Tim Middleton, Martin. B. Brilliant, Eyal Doron, John M. Merrill, Michael Tarley, Sheryl Coleman, Bryan Woodworth, Jonathan (J.N.)File, Darren Natale, Bill Gerrard, Matt Woodward, Dave Zangger, Yu-Han Ting,

Jouni A. Kosonen, Dale Nurden, Stephen Cowie and Carlos Rabaca.

Index

Introduction

Toolbar is intended to provide some features that I found lacking in the *Program Manager* and *File Manager* of the retail Windows 3.1. To start with, while working in *File Manager*, you can drag an item and drop it over some running programs. But if one of these programs is not running and you drop the item over its icon, the program is not started. Instead, the dropped item is added to the corresponding group of *Program manager*. Of course, sometimes this is just what you wanted. One could say that the groups in *Program manager* were meant more for organization of programs than for running them. On the other hand, most programs still lack the ability to accept dropped files and you cannot use with them this drag and drop feature.

Since my screen tends to be cluttered with windows, I also wanted the ability to place (in a permanent and compact way) at a corner of the screen the utilities I used most often. *Program Manager* groups are nice but big and I do not need to see the names of my favorite utilities: I can easily recognize them by their icons. The utilities should always be at hand - ready to be started by double-clicking them (as in *Program Manager*) or by dropping an item on them. For instance, this is a nice way to provide the most important facility missing in *File Manager*: a file viewer.

Toolbar is the answer to these wishes. It provides an alternative to the groups of *Program Manager* with the following distinctive characteristics:

> It can be used either as a substitute for *Program Manager* (which, in turn, can be run from *Toolbar*), or as a normal application installed in *Program Manager* (or in the current shell program).
>
> It is better suited for running programs: there are multiple ways of running programs and advanced settings can be used.

All programs in a toolbar are always visible (as long as the whole window is on the screen). They are held together in a very compact way, to be as unobtrusive as possible. Optionally, you can force a toolbar to be always visible on top of other windows. In order to save space, the title bar can be suppressed (or displayed to have useful information at hand).

Items from *File Manager* can be dropped on any program installed as a tool. It only needs to be able to accept file names on the command line. Most MS-DOS and Windows programs have this ability. A program can also be started by double-clicking (or single clicking the right button), or by using the enter key, or a user selected accelerator key.

You can choose between running an instance of the program for each dropped file, or running a single instance of the program for all of the dropped files.

Programs can be easily added, removed and reordered. The corresponding caption, command line, working directory, icon and other options can be easily changed.

You can implement a full tree of nested toolbars.

You can drag files to and from *Toolbar*. *Program Manager* does not allow dragging of its elements outside of its windows.

Utilities (small utility programs that use a toolbar button as its display window) make *Toolbar* more compact and extensible. A few are provided with *Toolbar*: trashcan, alarm clock, menu, memory, resources, information and blank icon. Additionally, new utilities can be added by the user (the programmatic interface is fully documented and sample source code is included).

Toolbar supports most of the same user interface as *Program Manager.*

Toolbar is able to transparently use (and to translate to its own format) *group* files from *Program Manager.*

Procedures

Installing *Toolbar*
Using nested toolbars
Running *Toolbar*
Installing elements
Selecting an element
Changing element properties
Arranging elements
Removing an element
Running an element
Using advanced options to run programs
Running programs
Changing *Toolbar* options
Using *Program Manager*'s group files
Using *Toolbar* instead of *Program Manager*
Saving options, commands and tools
Dragging elements
Getting help (See **Help**)
Reporting bugs (See **Suggestions and bugs**)
Writing utilities

Installing *Toolbar*

To install *Toolbar*:

1. Copy the executable (`toolbar.exe`) and the help file (`toolbar.hlp`) to the desired directory. To use the **Utilities**, copy the .`DLL` files as well.
   When *Toolbar* is run from within Windows:
      Upon execution, *Toolbar* will search for **`TOOLBAR.INI`** in the execution directory and create it if not found.
   When *Toolbar* is the shell:
      Upon execution, *Toolbar* will search for **`TOOLBAR.INI`** in the execution directory, then in the Windows directory. If not found, it will create it in the Windows directory.
2. You may want to install the program in a *Program Manager* group. You might even install it in the **`StartUp`** group to have *Toolbar* automatically started at the beginning of each Windows session.
3. You can use the dialog box of **Properties** (in *Program Manager*) to append to the command line the name of an **`INI`** file to be used instead of the default **`TOOLBAR.INI`**. In this way you can install and run simultaneously different copies of *Toolbar*. You can even install one or several trees of nested toolbars.

Note that to make easier using *Toolbar* from a network, if it is the shell and **`TOOLBAR.INI`** is not found in *Toolbar*'s directory, it will be searched for (or created) in the Windows directory.

Using nested toolbars

You can implement a full tree of toolbars simply by installing one or more secondary toolbars in the main toolbar. Each nested toolbar has its own **`INI`** file. Toolbars can be nested to any depth to give a very complete yet unobtrusive way of accessing your tools. You can even use *Toolbar* as the Windows shell by using the **Use as shell** command.

Running *Toolbar*

You can run *Toolbar* exactly in the same way as any other Windows program: by double-clicking on it, by using the **Run** menu entry of *Program Manager*, etc. If you find yourself using *Toolbar* on every Windows session, you can install it in the **`StartUp`** group to have *Toolbar* automatically started each time Windows starts.

The first time you run the program it will create the **`TOOLBAR.INI`** (or the **`INI`** file you selected in the installation process). In this file the program will automatically save information on the window location and appearance and on the installed elements. In this way, the next time you run the program it will start exactly at the same screen location and with the same elements and options.

If the program is running and you try to re-run it, the already running instance will be activated and its size restored if it was minimized. This can be used to help locate the program if it gets lost under other windows. You can only have a running instance of *Toolbar* for each **`INI`** file, because I cannot think of any advantage of having two exact copies of the program running at the same time. But, of course, by using different **`INI`** files, as described in the installation section, you can run simultaneously different toolbars. You can even nest them. (See **Using nested toolbars**.)

Note that to make easier using *Toolbar* from a network, if it is the shell and **`TOOLBAR.INI`** is not found in *Toolbar*'s directory, it will be searched for (or created) in the local Windows directory.

Installing elements

To add an item (a program or *tool*, a secondary toolbar or a utility) to the current toolbar use the **New element...** entry in the system menu of *Toolbar*, or press the ins key. (Alternately, if there is a space in the toolbar with no element, you can double-click there.) After having selected the type of element in the **New** dialog box, the **Tool properties**, **Toolbar properties** or **Utility properties** dialog box will appear to let you define the new element. You can also drag a file from the *File Manager* and drop it on the title bar of *Toolbar*, in which case the **Tool properties** dialog box will appear with default values for the **Title** and **Program** entries.

When you exit the **Tool properties**, **Toolbar properties** or **Utility properties** dialog box with **OK**, the new element will be installed and the *Toolbar* window will be resized to display it. You can change the appearance of this window by using the **Options...** menu entry.

You can also drag an element from one toolbar to another. It will be copied and no dialog box will be opened.

By selecting **Pick a window** the cursor changes and you can select with the mouse any window of a currently running program that you want to install in the toolbar. After pressing esc or selecting the desktop no tool is installed. No dialog box will be opened.

Selecting an element

To select an element, single click the element with the mouse or use arrow keys to change the selection. The current element has its title in a color background (if **Titles** is enabled) or appears in a red frame (if **Focus rectangle** is enabled and **Button like** is enabled or **Frame** is different from zero).

Once an element is selected you can:

Change its properties with alt+enter

Delete it with del
Run it with enter
Run it minimized with shift+enter

Changing element properties

To change the properties of an element, select it and use the **Properties...** entry in the system menu, or the alt+enter key combination (you may also click the right mouse button while holding down the alt key). The **Tool properties**, **Toolbar properties** or **Utility properties** dialog box will appear to let you customize the element.

Arranging elements

To change the position of an element, drag it over the desired new location and release the mouse. The button being moved will change to indicated the move operation.

There are several **Options** to control toolbar appearance, and a **Blank icon** can be inserted to leave a blank space for esthetic reasons. (e.g. to put different types of elements in each row).

It might be necessary to uncheck the **Drag and drop server** entry in order to be able to change the position of elements under Windows NT or other environments that do not support some undocumented Windows 3.1 drag-and-drop features.

Removing an element

To remove an element, select it and use the **Remove** entry in the system menu, or the del key. You will be prompted to confirm the deletion.

Running an element

To run an element, you simple use the mouse to double-click it (a single click on the right button will also work). You can also select it as described above and use the enter key. The third way to start an element is by using the **Fast load key** in the **Tool properties** dialog box.

The program will start minimized if you have chosen it as the default behavior in the **Tool properties** dialog box or if the shift key was held down when the program was invoked.

To start a tool with a data file, you drag the file from the *File Manager* and drop it over the tool icon.

Using advanced options to run programs

If this entry of **Tool properties** and **Toolbar properties** is checked, *Toolbar* will scan the **Program** command line (after having appended dropped files) to seek the following special options, *which are not case sensitive*:

1. If the **Multiple** instance option is checked, each dropped file will obtain its own copy of the tool and you must control the exact location and format in which the file is passed to the tool by using one or several of the following options:

   `%P` will be replaced by the complete file specification, including drive, directory, name and extension.

   `%F` will be replaced by the file specification with the extension (and the corresponding period) removed; i.e., it will include drive, directory and name.

   `%D` will be replaced by the file drive and directory. The trailing \ will be retained only in the case of the root directory.

   `%N` will be replaced by the file name, excluding drive, directory and extension.

   `%E` will be replaced by the file extension (without the starting period).

2. If the **Multiple** instance option is not checked, all dropped files will be appended with their complete file specification after the tool program. Nevertheless, you can place some options after all these files by using the following option:

   `%A` The remaining command line will be appended after the dropped files. This is useful when the program needs options after file names because, by default, names are appended to the end of the ***Program*** command line. It can appear only once.

3. Prompt the user for additional information:

   `%Q` A ***Toolbar is about to execute*** **dialog box** will be opened to let the user introduce additional options or change the current ones. This is useful when the program needs options that cannot be provided by drag and drop. It can appear only once and the command line will not be further scanned after this option. For instance, if you want to drop files over an archiving program called **`MyArc`**, you could use something like **`MyArc %Q`** to be prompted for the archive file name where the dropped filename must be added.

4. All other `%` options will be deleted except foe the following ones:

   `%%` will be replaced by a single `%`. This is useful if a program needs its own `%` options.

Running programs

You can run any DOS or Windows program from *Toolbar* (not just the installed tools as described in **Running an element**) Simply use the **Run...** entry in the system menu (or use the shift+r key combination). The ***Run*** **dialog box** will be displayed. You can also schedule invocation of a program by using the **Clock**.

Changing *Toolbar* options

To change *Toolbar* options use the **Options...** entry in the system menu, or the ctrl+o key combination. The **Toolbar options** dialog box will appear.

Using *Program Manager*'s groups

Instead of a normal **INI** file, you can specify a group file from *Program Manager* (they are usually placed in the Windows directory and have a .**GRP** extension). *Toolbar* will do its best to translate it. To make sure that *Toolbar* chooses the right number of columns and rows, select **Automatic** in the **Options** menu of *Program Manager*.

To save its configuration in the ordinary way, *Toolbar* will not use the group file, but an **INI** file with the same drive, path and name, but with the extension **INI**.

The next time the toolbar is started, it will read both the group file (to find the installed tools, the number of **Columns** and **Rows**, and the toolbar **Caption**) and the **INI** file (to find the remaining entries in **Options**).

Note that because the **GRP** file dictates the element order and **Properties**, changes must be made through *Program Manager* to be permanent. If you wish to permanently modify the toolbar, then replace the **GRP** file specification to use the **INI** file that is automatically created.

This is especially useful after a new program has been installed. Its group file can be readily converted to *Toolbar*'s **INI** format.

Using *Toolbar* instead of *Program Manager*

If you really like *Toolbar*, you could try using it as a replacement shell. To do that, simply check the **Use as shell** entry in the system menu.

To remove Toolbar as the shell and go back to using *Program Manager*, re-check the **Use as shell** entry in the system menu.

In either case, you will be given the chance to immediately restart Windows with the new shell. If you do not accept this (or if Windows does not actually exit because a DOS window is running or another application refuses to exit) the new shell will be used in the next Windows session.

Note that you can also install *Program Manager* as a tool to be started from *Toolbar*, much in the same manner as you can install *Toolbar* in a *Program Manager* group. Toolbars can be nested to any depth, unlike *Program Manager* groups (See **Using nested toolbars**). You may take advantage of the features of both programs.

The auto-start facilities of the **`StartUp`** group are provided by the file **`STARTUP.INI`** whose elements, after having being installed by *Toolbar*, will be automatically run when Windows starts and *Toolbar* is the shell. The entries **`[load]`** y **`[run]`** of **`WIN.INI`** are also correctly started.

Note that to make using *Toolbar* easier from a network, if it is the shell and **`TOOLBAR.INI`** is not found in *Toolbar*'s directory, it will be searched for (or created) in the Windows directory.

If you have problems using *Toolbar* as a shell, refer to **Solution to common problems**.

Saving options, commands and elements

To minimize disk access *Toolbar* usually saves the list of installed elements, the current options (if **Options** is On) and the last ten commands (if **History** is On) only when the program exits. If the Windows session is ended abruptly (because some program crashes, for instance) you will lose the changes performed during the current session. To avoid this, you can save the current configuration at any time by using the same trick that you can use in *Program Manager* and *File Manager*: simply try to close the program while holding down the shift key. shift+alt+f4 is an easy keyboard shortcut.

You can also uncheck the **Only at exit** option to have changes saved each time an option is changed or an element is added or removed.

Dragging elements

You can drag any element from one toolbar onto another toolbar (it will then be copied there) or, unlike from *Program Manager*, onto any another program accepting dropped files. The result of the latter action will depend on the program. Nevertheless, some programs (including *Program Manager* and *File Manager*) incorrectly assume that they are the only programs able to provide dragged files. This can make this feature of *Toolbar* less useful when used with other programs.

This feature is not available under Windows NT or other environment that does not support the undocumented drag-and-drop features of Windows 3.1.

Writing utilities

A user defined **utility** is a Windows **`DLL`** library that conforms to the protocol described in what follows:

1. It must export three **`CALLBACK`** functions by putting in the **`exports`** section of its **`def`** file the following lines:

```
EXPORTS                 InitToolbarDLL
                        FreeToolbarDLL
                        RunToolbarDLL
```

2. *Toolbar* loads the library and initializes it by calling:

```
lp = InitToolbarDLL(hInstance,lpParams);
```

The input parameters are:

HINSTANCE	**hInstance**	Calling instance
LPSTR	**lpParams**	Far pointer to parameters (**LENPARAM** bytes)

LENPARAM equals 256 and is the length, in bytes, of the block of instance specific memory provided by *Toolbar* to each copy of the library and pointed to by **lpParams**. As with any Windows **DLL** library, a utility has a single instance (a single **DATA** group in fact), but *Toolbar* provides each with a copy of the utility (installed in the same or in a different toolbar) with a private 256 byte long memory block which can be used as instance specific data. Furthermore, the first part of this block (up to the first null byte) will be saved by *Toolbar* as an ASCII string in the **INI** file and will be available in the next Windows (or toolbar) session.

You can take advantage of this feature by dividing (by means of a **struct**) this block in two parts:

> The first one contains the options that must be saved from session to session and must appear as printable ASCII characters with the exception of the comma(!) which you may not use. Integers may be represented in ASCII (decimal or hex), booleans as Y/N or T/F etc. This section must be null terminated.
>
> The second part may have whatever format you wish (binary, ASCII or mixed) and contain the instance specific variables that are not saved across sessions. For example, instance specific counters, times, etc.

The library has the opportunity to perform instance specific initialization tasks and must fill the information in a **InitDLLType** struct and return a long pointer to it or **NULL** on error (for instance, if your library cannot be run by two different instances or modules). The information returned by the utility must be a **struct** in the form:

```
typedef struct {
 BOOL fWantHDC;
 void (CALLBACK *lpTickToolbarDLL)(HWND hwnd,HDC hdc,LPSTR lpParams);
 void (CALLBACK *lpDrawToolbarDLL)(HWND hwnd,HDC hdc,LPSTR lpParams);
} InitDLLType;
```

fWantHDC must be **TRUE** if the library wants an **HDC** from *Toolbar* when called through **lpTickToolbarDLL** to process a **TIMER** message.

If the **lpTickToolbarDLL** is **NULL**, the library does not want to process **WS_TIMER** messages. Otherwise it must point to the **CALLBACK** function that must be called with the **Frequency** defined for the toolbar.

If the **lpDrawToolbarDLL** is **NULL**, the library does not want to process **WS_PAINT** messages. Otherwise it must point to the **CALLBACK** function that must be called every time its icon must be repainted.

3. *Toolbar* frees the library by calling:

```
FreeToolbarDLL(hInstance,lpParams);
```

The input parameters are the same as in the previous function and there is no return value.

4. If one or several files are dropped on the utility icon, they are passed in the form:

```
RunToolbarDLL(hwnd,lpParams,lpFiles);
```

where the input parameters are:

`HWND hwnd` Calling window handle. From which you can get the calling instance.
`LPSTR lpParams` Far pointer to parameters (see above)
`LPSTR lpFiles` Far pointer to an ASCII string with the dropped files separated by spaces.

5. If the utility is activated (by a double click, a single click on the right button or by pressing enter), the utility will receive the following call:

```
RunToolbarDLL(hwnd,lpParams,NULL);
```

where the input parameters are as in the previous function except for the fact that there is no dropped file.

6. If the utility answered to `InitToolbarDLL` that it intend to process `WS_TIMER` messages by setting `lpTickToolbarDLL` = `TickToolbarDLL`, timer ticks will be passed to in the form:

```
TickToolbarDLL(hwnd,hdc,lpParams);
```

where the input parameters are:

`HWND hwnd` Calling window handle. From which you can get the calling instance.
`HDC hdc` `HDC` with the clip rectangle to be drawn (or `NULL` if `fWantHDC` was `FALSE`).
`LPSTR lpParams` Far pointer to parameters (see above)

7. If the utility answered to `InitToolbarDLL` that it intend to process `WS_PAINT` messages by setting `lpDrawToolbarDLL` = `DrawToolbarDLL`, when the window must be repainted it receives:

```
DrawToolbarDLL(hwnd,hdc,lpParams);
```

where the input parameters are:

`HWND hwnd` Calling window handle. From which you can get the calling instance.
`HDC hdc` `HDC` with the clip rectangle to be drawn.
`LPSTR lpParams` Far pointer to parameters (see above)

8. The definitions needed to implement a utility are contained in the file `TOOLDLL.H` that may be included in your own programs. Moreover, you can use the `DEMODLL.*` files as a starting point - it is an implementation of a (nearly) useless but complete and fully commented example.

9. Your utility might be used for any program aware of the aforementioned protocol, even by users that do not use *Toolbar*. Such a program must cope with all the tasks done by *Toolbar*: allocating memory, loading, initializing and freeing the library, establishing handlers for `WS_TIMER` and `WS_PAINT` messages (as necessary), getting keyboard or mouse input, handling dropped files, calling the right library functions in the right way, etc. Sample source code is provided in `DEMODLL.*` which allows using `DEMODLL.DLL` as an standalone program that runs from its own icon (not from *Toolbar*). This is a small but complete program except for the fact that 1) no drag-and-drop handler is established (`DEMODLL.DLL` does not know what to do with dropped files) and 2) mouse clicks and keyboard enters are ignored (because, being an iconic utility, it is activated and `RunToolbarDLL` called

when a menu entry is used).

Commands

To save space, *Toolbar* does not have its own menu. It uses instead the system menu that can be accessed in the standard ways: typing alt+space or clicking at the symbol in the upper left corner of the window. The first method (or a **Menu icon**) works even if the symbol is not displayed because **Caption** has been suppressed.

After the standard entries (**Restore** , **Move**, etc.) it adds the following entries:

Entry	**Definition**	**Accelerator key**
New element...	Add a tool or toolbar	ins
Properties...	Change current element attributes	alt+enter
Remove	Delete the current element	del
Options...	Change program preferences	ctrl+o
Run...	Execute a program	ctrl+r
Use as shell	Use *toolbar* instead of *Program Manager*	
Help	Open help index window	f1
About Toolbar...	Version information	

New element...

This entry of the system menu (or the equivalent ins key) let you add a new tool or toolbar to the current toolbar. After having selected the type of element in **New**, the **Tool properties**, **Toolbar properties** or **Utility properties** dialog box will appear to let you define the new element.

Properties...

This entry of the system menu (or the equivalent alt+enter key combination) lets you customize the current element of the toolbar. A **Tool properties**, **Toolbar properties** or **Utility properties** dialog box will be opened.

Remove

This entry of the system menu (or the equivalent del key) lets you remove the current element from the current toolbar. You will be prompted to confirm the deletion.

Options...

This entry of the system menu (or the equivalent ctrl+o key combination) lets you define some properties of the current toolbar. An **Options** dialog box will be opened.

Run...

This entry of the system menu (which can also be accessed through the shift+r key combination) lets you run any DOS or Windows program from *Toolbar* (and not just the installed tools as described in **Running an element**). The ***Run* dialog box** will be displayed. The last ten commands are saved in an history list (and also across sessions if **History** is used).

Use as shell

You can access this entry in the system menu to install/uninstall *Toolbar* as a replacement shell.

You will be given the chance to immediately restart Windows with the new shell. If you do not respond yes (or if Windows does not actually exit because a DOS window is running or another application refuses to exit) the new shell will not be in effect until the next Windows session.

Note also that you can also install *Program Manager* as a tool to be started from *Toolbar*, much in the same manner as you can install *Toolbar* in a *Program Manager* group. Toolbars can be nested to any depth, unlike *Program Manager* groups (See **Using nested toolbars**). You may take advantage of the features of both programs.

The auto-start facilities provided by the **`StartUp`** group of *Program Manager* appear in *Toolbar* in the form of the **`STARTUP.INI`** file, whose elements, after having being installed by *Toolbar*, will be automatically run when Windows starts if *Toolbar* is the current shell. The entries **`[load]`** and **`[run]`** of **`WIN.INI`** are also correctly started.

If you have problems installing/uninstalling *Toolbar* as the shell, refer to **Solution to common problems**.

Shell

The program used to organize and launch other Windows programs. The default Windows shell is *Program Manager*, but it is possible to use another program (like *Toolbar*) for this job.

Help

You can access the *Toolbar* help index by using the **Help** entry in the system menu of *Toolbar*, or the f1 key. From a dialog box the **Help** button gives access to the corresponding help topic.

You may want to consult the **Using Help** topic.

About *Toolbar*...

This entry of the system menu contains the current version of *Toolbar* and Copyright information.

Dialog boxes

Toolbar uses the following dialog boxes

New element
Toolbar options
Tool properties and Toolbar properties
Utility properties
Select icon
Select program, Select INI file, Select library and Select icon file
Run
Toolbar configuration
Toolbar is about to execute

New element dialog box

This dialog box appears when installing a new tool, toolbar, or user provided utility (with **New element...** in the system menu, or the ins key). Apart from the usual **OK**, **Cancel** and **Help**, there are a group of radio buttons that let you select the kind of element to be added. If **OK** is selected and the element is a tool, a toolbar, or a user provided utility (but not one of the standard utilities), the corresponding dialog box (**Tool properties**, **Toolbar properties** or **Utility properties**) will be opened.

The following element types are available:

- **Tool** (any DOS or Windows program)
- **Toolbar** (see **Nested toolbars**)
- **Standard utilities:**

Trashcan
Alarm clock
Menu icon
Information icon
Memory icon
Resources icon
Blank icon

User provided utility

Furthermore, you can install any currently running program by pressing **Pick a window** and using the mouse to pick any of its windows. Using esc or selecting the desktop installs no tool.

Pick a window

This button of the **New element** dialog box lets you install any currently running program by pressing it and using the mouse to pick any of its windows. Using esc or selecting the desktop installs no tool.

Toolbar options dialog box

This dialog box is started by using the **Options...** entry in the system menu, or the ctrl+o key combination. Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, you can use the following items to customize the toolbar:

Caption: You may edit the title bar caption here and specify whether to display the title bar. Note that certain character pairs have special meaning.

View all: When checked, the toolbar dimensions are resize automatically whenever elements are added, deleted, or when the toolbar is resized.

Rows: Specify the number of element rows. If **View all** is selected **Columns** will be automatically changed to reserve space to display all elements.

First: The first row to display if **View all** is not selected.

Top window: If this entry is checked, the *Toolbar* window will be always visible over other windows on the screen (unless there are other "top" windows). If it is unchecked, the window can be covered by the active window in the usual way. Note that the property of being a top window is hereditary in Windows: child windows will have it and will be displayed over other windows. This can be a little inconvenient when displaying a help window.

Columns: The number of columns used to display elements. If **View all** is selected **Rows** will be automatically changed to reserve space to display all elements.

Frequency: Select the frequency, in seconds, at which the caption defined in the previous entry is refreshed to display the current values of the above defined information. If the frequency is negative, then the frequency is the absolute value, in milliseconds, of the specified frequency.

Border: If this option is selected, the toolbar will have a resizable border.

Frame: The pixel width of the frame to be drawn around each element. Valid values are 0 to 9 where 0 will display no frame.

Focus rectangle: If this entry is checked, a red rectangle will indicate the currently selected element when **Titles** is disabled and **Button like** is enabled or when **Frame** is not zero.

Horizontal scroll bar: When selected, the toolbar will have a horizontal scroll bar to allow scrolling through non-visible elements. Only meaningful if **View all** is not selected.

Vertical scroll bar: When selected, the toolbar will have a vertical scroll bar to allow scrolling through non-visible elements. Only meaningful if **View all** is not selected.

Small font: When checked, *Toolbar* will use a small but readable font to display tool captions (this allows display of longer titles). If unchecked, it will use the same font as *Program Manager*.

Titles: When checked, *Toolbar* will display a caption under each element. When uncheck, captions are only displayed for elements without associated icons. This feature is primarily to allow a more compact toolbar.

Button like: When checked, *Toolbar* will give the icons a standard button look.

Save icon position: When selected, causes Toolbar to remember the last place you put its icon (when minimized) and to place it there again. Otherwise the icon is placed by Windows.

Drag and drop server: When selected, Toolbar uses drag-and-drop to change the position of tools, to copy elements from one toolbar to another and to provide files to other programs. Otherwise only the first possibility is available and the cursor will change during the operation, but it will not be a ghost of the icon being moved.

Only at exit When checked, the toolbar configuration is save only upon exit. When unchecked, the configuration is saved after each modification (to prevent losing changes due to power failure or system lock-ups).

Other options may be also saved according to the state of the following two buttons.

Options: If this entry is checked, *Toolbar* will save the current options defined in this dialog box as the default to be used in the next session.

History: If this entry is checked, *Toolbar* will save the last ten commands issued in the ***Run* dialog box**. In this way they will be available in the next session.

After a tool is started, *Toolbar*'s look may change according to the state of the following four options:

Do nothing: Do nothing to *Toolbar* on activation of an element.

Minimize: Minimize *Toolbar* when one of its elements is activated. Useful when implementing a full tree of toolbars.

Quit: Exit *Toolbar* when of its elements is activated. Useful when implementing a full tree of toolbars.

Hide: Hide *Toolbar* when a button is activated. This option appears the same as the "quit" option except that successive runs of hidden toolbars are faster. This option is not selectable when the toolbar is the shell because the *Task Manager* does not show hidden tasks, which could lead to you not being able to access *Toolbar* once hidden. This could cause you not to be able to run any new programs or even exit Windows!

Caption

This ***Toolbar options* dialog box** option allows editing of the title bar caption and suppression of the entire title bar. Suppressing the title bar provides a more compact toolbar, while non-suppression allows easier movement of the toolbar and mouse access to the menu. The menu can always be accessed via the keyboard by pressing alt+space. From there you can move and resize the toolbar even with the title bar suppressed.

The caption can be any character sequence with the exception that certain character pairs are interpreted as special codes and can be used to save space by using the title bar to display useful information.

The pair	will be replaced by
`%d`	the current day (one or two digits)
`%D`	the current day (always two digits)
`%m`	the current month (one or two digits)
`%M`	the current month (always two digits)
`%y`	the current year (one or two digits)
`%Y`	the current year (always two digits)
`%h`	the current hour (one or two digits)
`%H`	the current hour (always two digits)
`%i`	the current minute (one or two digits)
`%I`	the current minute (always two digits)
`%s`	the current second (one or two digits)
`%S`	the current second (always two digits)
`%f`	the free memory in Mb
`%k`	the free memory in Kb
`%b`	the free memory in bytes
`%r`	the free resources percentage

For instance, **`%h:%I - %fMb - %r%`** will display a string in the title bar in the form **`1:02 - 3Mb - 71%`** indicating the current time, free memory and free resources.

The update rate of this information is defined in **Frequency**.

Information displayed in this way can also be obtained through an **Information icon** or by using the following utilities (which provide additional features as well):

Alarm clock
Free memory
Free resources

View all

When selected, this option of the ***Toolbar options*** **dialog box** causes the toolbar to automatically resize, so that all elements are always visible, whenever the number of elements changes or the toolbar is resized.

Rows

This option of the ***Toolbar options*** **dialog box** controls the number of rows used to display the elements. If **View all** is selected **Columns** will be automatically changed to reserve space to display all the elements.

This value may be also changed by using the mouse on an optional **Border**.

First

This options of the ***Toolbar options* dialog box** controls which is the first row display (if **View all** is not selected).

It may also be changed by using pg up and pg down on an optional **Vertical** or **Horizontal scroll bar**.

Top window

When checked, this option of the ***Toolbar options* dialog box** causes the toolbar to always be visible over an other windows on the screen (unless those windows are "top" windows as well). If unchecked, the window can be covered by the active window in the usual way. Note that the property of being a top window is hereditary in Windows: child windows will have it and will be displayed over other windows. This can be a little inconvenient when displaying a help window.

Columns

This options of the ***Toolbar options* dialog box** controls the number of columns used to display the elements. If **View all** is selected **Rows** will be automatically changed to reserve space to display all the elements.

This value may be also changed by using the mouse on an optional **Border**.

Frequency

In this entry of the ***Toolbar options* dialog box** you can select the frequency, in seconds, at which the **Caption** in the *Toolbar* title bar is refreshed to display the current values of the displayed information.
This option of the ***Toolbar options* dialog box** controls the frequency, in seconds, at which the caption defined in the previous entry is refreshed to display the current values of the above defined information. If the frequency is negative, then the frequency is the absolute value, in milliseconds, of the specified frequency.

Certain **Utilities** also rely on this frequency.

Border

If checked, this option of the ***Toolbar options* dialog box** adds a resizable border to the toolbar.

This option is different from **Frame**, which refers to each element window.

Frame

This option of the ***Toolbar options* dialog box** controls the pixel width of the frame to be drawn around each element. Valid values are 0 to 9 where 0 will display no frame.

See **Button like** and **Border** which refer to window of the whole toolbar.

Focus rectangle

When checked, this option of the ***Toolbar options* dialog box** outlines the currently selected element with a red line when **Titles** is disabled and **Button like** is enabled or when **Frame** is not zero.

This aids in supporting the keyboard interface.

Horizontal scroll bar

When checked, this option of the ***Toolbar options* dialog box** will add a horizontal scroll bar that allows scrolling through non-visible elements of the toolbar. This option is only meaningful when **View all** is not selected. The same effect can be had by using the pg up and pg down keys.

Vertical scroll bar

When checked, this option of the ***Toolbar options* dialog box** will add a vertical scroll bar that allows scrolling through non-visible elements of the toolbar. This option is only meaningful when **View all** is not selected. The same effect can be had by using the pg up and pg down keys.

Small font

When checked, this option of the ***Toolbar options* dialog box** causes *Toolbar* to use a small but very readable font to display element captions (this allows display of longer titles). When unchecked, it will use the same font as *Program Manager*.

Titles

When checked, this option of the ***Toolbar options* dialog box** causes *Toolbar* to display a caption under each element. When uncheck, captions are only displayed for elements without associated icons. This feature is primarily to allow a more compact toolbar.

Save icon position

This option of the ***Toolbar options*** **dialog box** lets you select if *Toolbar* should remember its icon position and place it in the same location when it is minimized again. This feature provides you more control over the desktop appearance, but some icons may disappear if they are moved out of the visible screen when the graphics resolution is lowered. (To recover the missing icon press ctrl+esc, or double click at the desktop, to open the *Task list*; then select **Arrange icons**.)

If this entry is not checked, Windows places the icon in the next free 'icon slot'.

Button like

When checked, this option of the ***Toolbar options*** **dialog box** causes *Toolbar* to give the icons a standard button look. (You must still double-click the button to active it)

See also **Frame**

Drag and drop server

This option of the ***Toolbar options*** **dialog box** lets you select if *Toolbar* should use true drag-and-drop to change the position of tools, to copy elements from one toolbar to another and to provide files to other programs. If it is not selected, only the first possibility is available and the cursor will change during the operation, but it will not be a ghost of the icon being moved. It might be necessary to uncheck this entry in order to be able to change the position of elements under Windows NT or other environments that do not support some undocumented Windows 3.1 drag-and-drop features.

Only at exit

When checked, this option of the ***Toolbar options*** **dialog box** causes toolbar to save the toolbar configuration only upon exit. When unchecked, the configuration is saved after each modification (to prevent losing changes due to power failure or system lock-ups).

Options

When checked, this option of the ***Toolbar options*** **dialog box** will save the current options selected in this dialog box as the default to be used in the next session. When unchecked, only the installed elements are saved to the **`INI`** file.

History

When checked, this option of the ***Toolbar options*** **dialog box** will save the last ten commands issued in the ***Run*** **dialog box**. In this way they will be available in the next session.

Do nothing

When selected, this option of the ***Toolbar options*** **dialog box** specifies that no action be taken when one of its elements is activated.

Minimize

When selected, this option of the ***Toolbar options*** **dialog box** causes the toolbar to minimize (shrink to an icon) when one of its elements is activated. Useful when implementing a full tree of toolbars.

See **Procedures**

Hide

When selected, this option of the ***Toolbar options*** **dialog box** causes the toolbar to hide when one of its elements is activated. To re-display, run the toolbar. This option is the same as the **Quit** option except that successive runs of hidden toolbars are faster than with **Quit**. Useful when implementing a full tree of toolbars.

This option is not selectable when the toolbar is the shell because the *Task Manager* does not show hidden tasks, which could lead to you not being able to access *Toolbar* once hidden. This could cause you not to be able to run any new programs or even exit Windows!

See **Procedures**

Quit

When selected, this option of the ***Toolbar options*** **dialog box** causes the toolbar to exit when one of its elements is activated. Useful when implementing a full tree of toolbars.

See **Procedures**

Tool properties and *Toolbar properties* dialog boxes

These dialog boxes are started when installing a new tool or toolbar (by using the **New element...** entry in the system menu, or the ins key) or customizing the selected tool or toolbar (by using the **Properties...** entry in the system menu, or the alt+enter key combination). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, you can use the following items to change the attributes of the element:

Title: The caption that will optionally be displayed to identify the element. **If you do not provide any title, the element will remain installed only during the current session of *Toolbar*!** (This can be useful to install a tool that will be used only during the current session.)

Program: This entry appears only in the *Tool properties* dialog box and contains the complete file specification of the program to be run, including optional parameters. You can directly enter the program specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select program** dialog box that will be displayed. Instead of an executable file you can also specify any (document) file whose extension is associated to an executable file. For instance, using `file.wri` you will actually get `write file.wri`; i.e., the `write.exe` program with the `file.wri` file as parameter.

File: This entry only appears in the *Toolbar properties* dialog box and contains the complete file specification of the `INI` file used by the nested toolbar. You can directly enter the file specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select INI file** dialog box that will be displayed.

Directory: The directory in which the program will be started. This entry is optional but highly recommended because many programs do not run properly if they are not started from the right directory.

Icon: This displays the icon that will represent the element in the toolbar. By default, the first icon in the executable file will be used, but you can select another icon by using the **Change...** button and the **Select Icon** dialog box that will be displayed.

Minimized: If the entry is checked, the element will start minimized. If it has not been selected, you can still run the element minimized if the shift key is held down when the program is invoked.

Multiple: This entry controls the behavior of the tool when several files are simultaneously dropped on it. If the entry is checked, each dropped file will start its own instance of the program. If it is not checked, all the dropped files will be appended to the command line of a single instance of the program. Note that whatever this setting is, if different files are dropped at different times, each of them will get its own copy of the tool.

Fast load key: This optional entry lets you select an accelerator key that will start the element.

Advanced options: This entry lets you specify if you want to use advanced options in the **Program** command line. See **Using advanced options to run programs**.

Exactly as in *Program Manager*, when an MS-DOS application is minimized the icon selected in this dialog will be used, but Windows applications have their own icon displayed. However, there is an exception when using a toolbar that has been installed as an element of another toolbar. Though it is a Windows program, if an icon has been selected in the aforementioned **Icon** entry, it will be used instead of *Toolbar*'s icon and it will be displayed with a special look: as a button. A primary toolbar (i.e., a toolbar that has not been installed inside another toolbar) will use *Toolbar*'s icon, but a power user can change this by adding to the corresponding `INI` file two lines in the form:

```
Icon file=file
Icon number=n
```

where ***`file`*** is the complete specification of the file containing the icon and ***`n`*** gives the order number (starting at 0) of the icon in the file. The second line can be omitted if the first icon will be used.

Utility properties dialog box

This dialog box is started when installing a new user utility (by using the **New element...** entry in the system menu, or the ins key) or customizing the selected utility (by using the **Properties...** entry in the system menu, or the alt+enter key combination). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, you can use the following items to change the attributes of the element:

Title: The caption that will optionally be displayed to identify the element. **If you do not provide any title, the element will remain installed only during the current session of *Toolbar*!** (This can be useful to install a utility that will be used only during the current session.)

Library: This entry contains the complete file specification of the utility library. It is automatically filled in for utilities provided with Toolbar. In the case of user provided libraries, you can enter directly the library specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select library** dialog box that will be displayed.

Icon: This displays the icon that will represent the utility in the toolbar. By default, the first icon in the executable file will be used, but you can select another icon by using the **Change...** button and the **Select Icon** dialog box that will be displayed.

Fast load key: This optional entry lets you select an accelerator key that will activate the utility.

Title

This entry of **Tool properties**, **Toolbar properties** and **Utility properties** contains the caption that will optionally be displayed to identify the element. **If you do not provide any title, the element will remain installed only during the current session of *Toolbar*!** (This can be useful to install a tool that will be used only during the current session.)

Since the space used to display the title is so narrow, you can take advantage of the possibility of using two lines. Simply use the pair **`\n`** to indicate the line break. For example, **`One\nTwo`** will display **`One`** in the first line and **`Two`** in the second one.

Program

This entry of **Tool properties** contains the complete file specification of the program to be run, including optional parameters. You can directly enter the program specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select program** dialog box that will be displayed.

Instead of an executable file you can also specify any (document) file whose extension is associated with an executable file. For instance, using `file.wri` you will actually get `write file.wri`; i.e., the `write.exe` program with the `file.wri` file as a parameter.

You can also use advanced options (see **Using advanced options to run programs**).

File

This entry of ***Toolbar properties*** contains the complete file specification of the `INI` file for the options and installed elements of the nested toolbar. You can directly enter the file specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select INI file** dialog box that will be displayed.

Library

This entry of **Tool properties** contains the complete file specification of the program to be run, including optional parameters. You can directly enter the program specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select program** dialog box that will be displayed.

Instead of an executable file you can also specify any (document) file whose extension is associated with an executable file. For instance, using `file.wri` you will actually get `write file.wri`; i.e., the `write.exe` program with the `file.wri` file as a parameter.

You can also use advanced options (see **Using advanced options to run programs**).

Browse...

This button of the **Tool properties**, **Toolbar properties**, **Library properties** and **Select icon** dialog boxes lets you search through your disks and directory tree for a file containing a tool or an icon. A **Select program**, **Select INI file** , **Select**

library or **Select icon file** will be opened.

Directory

This entry of **Tool properties** and **Toolbar properties** contains the directory in which the program will be started. Although this entry is optional you should consider specifying it, because many programs will only work properly if they started from a specific directory (usually their own directory).

Change

By default, the first icon in the executable file **Program** will be used to represent the element in the toolbar, but you can select a specific icon by using this button of **Tool properties**, **Toolbar properties** or **Library properties** and the **Select Icon** dialog box that will be displayed.

Minimized

If the entry of **Tool properties** or **Toolbar properties** is checked, the element will start minimized. If it has not been selected, you can still run the element minimized if the shift key is held down when the program is invoked.

Multiple

This entry of **Tool properties** or **Toolbar properties** controls the behavior of the element when several files are simultaneously dropped on it. If the entry is checked, each dropped file will start its own instance of the program. If it is not checked, all the dropped files will be appended to the command line of a single instance of the program. Note that whatever this setting is, if different files are dropped at different times, each of them will get its own copy of the tool.

Remember also that MS-DOS programs have very short command lines (less than 128 characters). As a consequence, they can receive only some of the dropped files and the last received file can have its file name truncated.

Fast load key

This optional entry of **Tool properties**, **Toolbar properties** or **Library properties** lets you select an accelerator key that will start the element.

Advanced options

This entry of **Tool properties** and **Toolbar properties** lets you specify whether you want to use advanced options in the **Program** command line. See **Using advanced options to run programs**.

Select icon dialog box

This dialog box is started by using the **Change...** button in the **Tool properties**, **Toolbar properties** and **Utility properties** dialog boxes. Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, you can use the following items to change the element icon:

> **File name**: The complete file specification of the file containing the icon. The default is the program file but you can directly enter a new file specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select icon file** dialog box that will be displayed.
>
> **Current icon**: This list box lets you select the icon to be associated with the element. You can also choose to have no icon.

File name

This entry of the ***Select icon* dialog box** contains the complete file specification of the file containing the icon. The default is the program file but you can directly enter a new file specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select icon file** dialog box that will be displayed.

Current icon

This list box of the ***Select icon* dialog box** lets you select the icon to be associated with the element. You can also choose to have no icon.

Select program, *Select INI file*, *Select library* and *Select icon file* dialog boxes

These dialog boxes are invoked but not provided by *Toolbar*. They are four instances of the common dialog box that most Windows 3.1 programs use to find and open a file. If you are not familiar with them and you really, really can't figure it out, then you probably shouldn't be using a computer! (you can find a description in your Windows manual)

Run dialog box

This dialog box is started by using the **Run...** entry in the system menu or by using the ctrl+r accelerator key. Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, you can use the following items to select the program to be started:

Command line: The complete file specification of the executable file including optional parameters if necessary. You can directly enter a new file specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select program** dialog box that will be displayed. Instead of an executable file you can also specify any (document) file whose extension is associated to an executable file. For instance, using `file.wri` you will actually get `write file.wri`; i.e., the `write.exe` program with the `file.wri` file as a parameter. This entry also contains a list of the last ten commands.

Run minimized: Check this button if you want the program started as an icon.

Save command in history list: Uncheck this button if you do not want the command saved in the history list.

Command line

This edit box of the ***Run* dialog box** lets you enter the complete file specification of the executable file including optional parameters if necessary. You can directly enter a new file specification or search for it through your disks and directory tree by using the **Browse...** button and the **Select program** dialog box that will be displayed.

Instead of an executable file you can also specify any (document) file whose extension is associated with an executable file. For instance, using `file.wri` you will actually get `write file.wri`; i.e., the `write.exe` program with the `file.wri` file as parameter.

This entry also contains a list of the last ten commands. They are also available across sessions if **History** is used. See also **Save command in history list**.

Run minimized

This check box of the **Run dialog box** lets you specify if the program to be launched should start reduced to an icon.

Save command in history list

This check box of the ***Run* dialog box** lets you specify if the program to be launched should be saved in the list of the last ten commands. See also **History**.

Toolbar configuration dialog box

This dialog box may appear when *Toolbar* is using a *Program Manager*'s group file to indicate that the `INI` file it will use already exists and does not correspond to the same toolbar. You are given the choice of overwriting it (pressing **OK**) or of using no `INI` file.

Toolbar is about to execute dialog box

This dialog box is started when a tool having the **Advanced options** property set in **Tool properties** and the `%Q` option has been included in the **Program** command line. You will be prompted with the actual command line that will be executed (including dropped files). The cursor will be at the position indicated by `%Q`. You can then edit the line to add or change options. If **OK** is pressed the command line will be immediately executed. You can also choose **Cancel** to abort the procedure and **Help** to obtain help.

See also **Using advanced options to run programs**.

Utilities

Icon utilities have been added to *Toolbar* (mainly as a consequence of contradictory suggestions from users) to take into account the following points:

There exist a lot of little utilities (trashcans, clocks, memory or resources information, etc.) that run nearly exclusively from its own icon. Since after installing them in a toolbar, a copy of the icon will be present, it seems better to use it instead of a second copy.

Toolbar uses system resources that other programs could exploit. For instance, if **Frequency** is non-zero, the toolbar will start a timer (maybe to refresh the information in the **Caption**). It would be great using the same timer to accomplish other tasks, such as scheduling the launching of a program at a later time.

Furthermore, utilities have some advantages:

They allow to see the information displayed in the **Caption** even when the latter is suppressed. In the same case, they allow accessing the menu by means of the mouse.

They can be used by other programs, even if *Toolbar* is not running.

They provide a practical way to keep the conservative use of system resources that some users expect from the shell while optionally providing the features that other users want. Each user choose the utilities to be started. The user can even add his (or her) own utilities.

By using utilities, *Toolbar* can be extended and persons other than its author can directly contribute to its improvement. It would be great if we had free contributions from other people!

They can be easily written, because part of the job is done by *Toolbar*. The source code for a complete example is included. It can be used as the starting point for more useful utilities. The example also shows how to write programs which are able to use these utilities even if *Toolbar* is not running.

The following standard utilities are included with the current version of *Toolbar*:

Trashcan
Menu icon
Information icon
Alarm clock

Free memory
Free resources
Blank icon
Exit toolbar icon
Exit Windows icon

Moreover, **User utilities** can be added.

Trashcan

This utility provides an alternative way of deleting files from the *File Manager*. Install it in one or several toolbars by using ins o **New element...** in such a way that it can be easily accessed (remember the option **Top window**). If you drag a file or group of files from *File Manager* and drop it over the trashcan icon, it will be deleted. If the icon looks like a garbage can, the erased files can be easily undeleted by using this utility, because in fact they are only moved to a hidden directory. (A trashcan icon is used if there are erased files in that directory and a different one if it is empty.) If the icon looks like a shredder, the only way to unerase them is to use the DOS `Undelete` or a similar utility.

The trashcan is controlled from the dialog box which opens by double clicking on the utility icon (or single-clicking the right mouse button). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, there are the following entries:

Confirm deleting Here you choose if you want to be asked for confirmation before deleting (or moving to the trashcan):

Every file	(whatever its type is)
Special files	(Hidden, System and Read only files)

If none of these options is selected, files will be quietly deleted/moved.

Empty trashcan You select here if you want all files in the trashcan removed (in such a way that they could only be unerased by `Undelete`) when:

At exit	(The trashcan will be emptied before exiting)
Immediately	(It will be a shredder, instead of a trashcan)

If none of these options is selected, the deleted files will remain in the trashcan, even if the computer is turned off. (You can restore them at any time, but in the meanwhile they use disk space.)

Deleted files This is the list of deleted files stored in the trashcan. You can select one or more of them and select:

Undelete	they will be restored to their original directory.
Remove	they will be deleted from the trashcan.
Remove all	the trashcan will be emptied.

Trashcan directory This is the complete specification of the hidden directory that will contain the deleted files. If you try changing it, a message box will prompt you to confirm the change. You may prefer to empty the directory with **Remove all** before changing it.

All changes and actions selected from this dialog box will actually be done when (and only if) you choose **OK**. They will be ignored if you use **Cancel**.

You can install more than one trashcan. This is be especially useful when one is a shredder and the other a true trashcan or if they have trashcan directories in different disk units, because moving a file is faster when the source and destination disks are the same. If two trashcans have the same directory, they have full access to it and what is added or removed by one of them will be seen in the same way by the other. So, if one copy empties the trashcan (maybe because it is closed) the other will find nothing left; but since it is not informed its icon may indicate that there are still erased files (the icon will be updated after opening the dialog box).

System menu icon

If the **Caption** is suppressed the system menu icon is not displayed at the left top corner of the toolbar, although it can still be accessed by pressing alt+space. If you prefer to use the mouse, install this utility (using ins or **New element...**); then double click it (or single click the right mouse button) to open the menu. This gives access to the commands to move and size the window, to end and switch tasks and to *Toolbar*'s **Commands**.

Information icon

If the **Caption** is suppressed, you loose the ability to use it to display useful information. To retain this ability, install this utility (using ins or **New element...**). The utility icon is able to display up to three lines with information controlled from the dialog box which opens by double clicking on the utility icon (or single clicking the right mouse button). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, the **Format string** edit control lets you enter the information to be displayed, with the following pairs of characters having special meaning:

The pair	**will be replaced by**
%d	the current day (one or two digits)
%D	the current day (always two digits)
%m	the current month (one or two digits)
%M	the current month (always two digits)
%y	the current year (one or two digits)
%Y	the current year (always two digits)
%h	the current hour (one or two digits)
%H	the current hour (always two digits)
%i	the current minute (one or two digits)
%I	the current minute (always two digits)
%s	the current second (one or two digits)
%S	the current second (always two digits)
%f	the free memory in Mb
%k	the free memory in Kb
%b	the free memory in bytes
%r	the free resources percentage

For instance,

```
%h:%I
%fMb
%r%
```

will display as three lines in the form:

```
 1:02
 3Mb
71%
```

indicating the current time, free memory and free resources.

The update rate of this information is defined in **Frequency**, which should be non-zero.

The different types of information which this utility may display can also be obtained (separately but with added features) by using the following utilities:

Alarm clock
Free memory
Free resources

Clock

This utility is an alarm clock with the additionally capability of scheduling the automatic activation of a program when the alarm goes off.

It can be installed by using ins or **New element...**). Then, its icon can display the current time (in different formats) and, eventually, a little bell symbol to indicate that the alarm is set.

The clock is controlled from the dialog box which is activated by double clicking on the utility (or single clicking the right mouse button). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, the following entries are available:

Display The display format can be:

Analog (analog clock)
12 hours digital (digital in American format)
24 hours digital (digital in European format)

Alarm :

On If the alarm is set.
Beep If beeps are desired.
Time The alarm time in 24 hour format: 00:00 to 23:59.
Execute The program to be activated (or nothing). Its complete specification can be directly entered or by using...
Browse to search for it through disk and directories.

If the alarm is **On**, when it goes off:

If **Beep** is set, the icon will start beeping until you double click (or single click the right button) it. (You can also select it with the keyboard and press enter.)
The program in **Execute** (if any), will be started.

The clock will only work if **Frequency** is non-zero, because it controls the time refresh rate.

Free memory information

This utility, which can be installed by using ins or **New element...**, displays in its icon the current free memory in different formats.

It is controlled from the dialog box which is activated by double clicking on the utility (or double clicking the right mouse button). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, the following entries are available:

Display The display format can be:

Megabytes	(1 Mb = 1024 Kb)
Kilobytes	(1 Kb = 1024 bytes)
Bytes	(It displays in three lines **m**, **k**, and **b** to indicate that the free memory is **m** Mb + **k** Kb + **b** bytes = (**b** + 1024 (**k** + 1024 **m**)) bytes)

Small letter . This should be selected (unless **Megabytes** was chosen) to have room to display the information.

This utility will only work if **Frequency** is non-zero, because it controls the refresh rate.

Free resources information

This utility, which can be installed by using ins or **New element...**, displays in its icon the current free resources in graphic and numeric formats.

It is controlled from the dialog box which is activated by double clicking the utility (or single clicking the right mouse button). Apart from the self-explanatory **OK**, **Cancel** and **Help** buttons, the following entries are available in **Display**:

Graphics resources	(in the GDI segment)
Other resources	(in the USER segment)
Minimum of both	

This utility will only work if **Frequency** is non-zero, because it controls the refresh rate.

Exit toolbar button

This utility, which can be installed by using ins or **New element...**, closes the toolbar when activated. So, it is equivalent to pressing alt+f4.

If the toolbar is the shell, you will be prompted to confirm exiting Windows. (For a quicker way to exit Windows, use a **Exit Windows button**.

Exit Windows button

This utility, which can be installed by using ins or **New element...**, closes the current Windows session (unless an application refuses to end or there is any DOS window). **No confirmation is asked**.

See also **Exit toolbar button**.

Blank button

This utility, which can be installed by using ins or **New element...**, does nothing, except to fill a toolbar location. It may be used to arrange the icons in the toolbar (to have a different type of element in each row for instance).

A double click on the utility (or single clicking the right mouse button) will open this help window.

User utility

A user provided utility is installed by using ins or **New element...** and choosing in a ***Select library*** **dialog box** the corresponding **DLL** library. Note that you cannot specify an arbitrary Windows **DLL** library; if it does not conform to *Toolbar*'s protocol for utilities, a message error will be issued and it will be of no use in the toolbar. The standard utilities can also be installed in this way, although you can directly select them from the ***New element*** **dialog box**.

By using **Utilities**, *Toolbar* can be extended and persons other than its author can directly contribute to its improvement. It would be great if we had free contributions from other people!

They can be easily written, because part of the job is done by *Toolbar*. The source code for a complete example is included. It can be used as the starting point for more useful utilities. The example also shows how to write programs which are able to use these utilities even if *Toolbar* is not running. (See **Writing utilities**.)

Error and warning messages

The following messages are issued by *Toolbar* (or its *Trashcan*) when necessary:

`Cannot run another instance`

A tool refused to start because only one instance of the program (or a necessary library) can be run.

`Could not create a window`

Improbable error that indicates that a window could not be created (perhaps by lack of memory).

`Could not create directory` *(Trashcan)*
The trashcan was unable to create a directory.

`Could not create the file` *(Trashcan)*
The trashcan was unable to create a file.

`Could not erase this file` *(Trashcan)*
The trashcan was unable to delete a file.

`Could not move the file` *(Trashcan)*
The trashcan was unable to move a file.

`Could not open the file` *(Trashcan)*
The trashcan was unable to open a file.

`Could not open the index file` *(Trashcan)*
The trashcan was unable to open the file with the list of deleted files.

`Could not read the index file` *(Trashcan)*
The trashcan was unable to read the file with the list of deleted files.

`Could not register`
Improbable error that happens if it was not possible to add *Toolbar* window class to the Windows database.

`Could not run the program`
A tool refused to start. Perhaps there was not enough memory or the executable file was missing or incorrect.

`Could not save the index file` *(Trashcan)*
The trashcan was unable to save the file with the list of deleted files.

`Exit Windows?`
If your answer is affirmative, Windows session will finish. It is only possible if *Toolbar* is the current shell.

`File not found`
A tool refused to start because the specified file has not been found.

`Incompatible .DLL library`
The user utility does not conform to the protocol for utilities.

`Incorrect Windows version or mode`
A tool refused to start because it was created for another version of Windows or because it cannot be run in the current (standard or enhanced) mode.

`Invalid program file`
A tool refused to start because the program file was corrupt or was created for another

operating system (OS/2, for instance).

`Missing or invalid file`
An **`INI`** file must be specified for each toolbar installed as a tool inside another toolbar.

`No association defined for this extension`
A tool refused to start because the specified file has an extension that is not associated with any executable program. Specify the latter or define the association in *File Manager*.

`No program defined for this tool`
A tool refused to start because you have not specified the file to be run.

`No timer available`
A resource necessary to run *Toolbar* correctly is exhausted.

`No trashcan directory` *(Trashcan)*
If the trashcan directory name is deleted, you can only use it as shredder.

`Not enough memory`
There is not enough memory to pass files to a tool (or to perform other tasks). Try closing other applications.

`OK to change trashcan directory?` *(Trashcan)*
Before changing it you should consider removing or unerasing deleted files.

`OK to delete directory?` *(Trashcan)*
The trashcan asks for permission to delete a directory.

`OK to delete special file?` *(Trashcan)*
The trashcan ask for permission to delete a system, hidden or read-only file.

`OK to delete?` *(Trashcan)*
The trashcan ask for permission to delete a file.

`Path not found`
A tool refused to start because the path of the specified file has not been found.

`Remove` ***`element name`*****`?`**
or
`Remove the current element?`
Last opportunity before deleting a tool from the toolbar.

`Restart Windows now?`
If your answer is affirmative, Windows will restart with the new shell.

`This element has no name`
`It will not be available`

`in the next Toolbar session`
Warning reminding you what will happen with nameless tools.

`This file has no icon`
`Use icons in Program Manager?`
If your answer is affirmative, *Toolbar* will show you the icons in **`PROGMAN.EXE`**.

`Too many deleted files,`
`only some of them are shown` *(Trashcan)*
There are too many deleted files. To see the remaining ones, you should remove or undelete some files and then restart the trashcan.

`Toolbar has no element`
`To add an element, press Ins`
`To get help, F1`
Warning indicating that the current toolbar is void.

`Two elements have now the same name`
`Only one of them will be available`
`in the next Toolbar session`
Warning reminding you what will happen if tool names are repeated.

Solutions to common problems

If you try to drag an item from *Program manager*, it will not work. (This is a limitation of *Program Manager*, not of *Toolbar*.) You must drag from *File Manager*. (There might exist other programs that are able to operate as drag-and-drop servers, as *Toolbar* does.)

If when minimizing a toolbar it disappears, it is probably located outside the visible screen because the graphics resolution has been lowered and the option **Save icon position** was On. To display the missing icon, open the *Task list* (by pressing ctrl+esc, or double clicking the desktop) and then select **Arrange icons**.

If a utility (the clock, for instance) does nothing, it may need a non-zero value for **Frequency**.

If you have problems installing/uninstalling *Toolbar* as the shell , a description of how all this works might be helpful:

1. The shell is the program indicated in the entry **`shell`** of **`SYSTEM.INI`** when the Windows session starts (and only then).
2. If *Toolbar* is told to be the shell (for the next session, of course), it copies the value of **`shell`** to the entry **`Previous shell`** of **`TOOLBAR.INI`**. Then it writes **`shell`** with its own specification (**`C:\toolbar\toolbar.exe`**, or something similar).
3. If *Toolbar* is told not to be the shell in the next session, it restores **`shell`** to the value saved in **`Previous shell`**.
4. You can always recover by hand the previous shell by writing **`shell`** with a specification (**`PROGMAN.EXE`** will work for *Program Manager*) by using any text editor (**`notepad`**, for instance).

If you experience problems when minimizing non-Windows programs and are using a very low (CGA, EGA) or very high (8514, etc.) resolution card, try adding to the `[Options]` section of the `INI` file an entry of the form:

```
Use DDE=0
```

If you have chosen a **Border** and a blank space appears to the right or the vertical scroll bar is not correctly hidden or displayed the first time, you are not seeing a bug in *Toolbar*. The blank space appears because Windows does not allow a window with a border to have a width below a certain value. To avoid the blank space you must have at least three **Columns** or two columns and a **Vertical scroll bar**. To have the vertical scroll bar correctly displayed or hidden, change the window size (or minimize and restore it).

If you are using a ATI Wonder card and *Toolbar* is the shell and its caption is not displayed, you may be experiencing problems with the first DOS window. You may get the ATI drivers for Windows 3.1, or start the first DOS window in full screen mode, or reset the video mode from the DOS prompt.

To move a window whose **Caption** has been suppressed, activate the system menu by typing alt+space (or using a **Menu icon**), press an arrow key and move the window with the mouse or the arrow keys. If there is some space in the toolbar with no icon, you can also drag it with the mouse.

If *Toolbar* does not close the Windows session when it is the shell and exits, try adding to the `[Options]` section of the `INI` file an entry of the form:

```
Exit Windows=1
```

If you cannot drag the elements in order to change their order or to copy them to another toolbar, you are probably running Windows NT or other environment that does not support the undocumented drag-and-drop features of Windows 3.1. You must disable them by unchecking the **Drag and drop server** option. Then, you can only change the order of elements.

Suggestions and bugs

The Author would gratefully welcome information about any bug or problem found in the program. Suggestions to improve *Toolbar* in future versions are also welcome. Write to:

Juan M. Aguirregabiria
Fisika Teorikoa. Zientzi Fakultatea
Euskal Herriko Unibertsitatea
P.K. 644, 48080 Bilbo (Spain)
Internet address: **wtpagagj@lg.ehu.es**