MENÚ BOOT ver.1.4 ---------- By Javier Guerra

Content

Description

The 'BOOT MENU' program is designed for use on disks and/or microdrives that contain several programs on the same medium and that require different BOOT or EXEC files to be executed. It is therefore used to facilitate the selection of programs by means of a menu once the support has been inserted into the reader.

In short, the 'BOOT MENU' is nothing more than an interactive BOOT of BOOTs.

It's also useful for running an application that supports the use of different data files or for anything that can be launched from a BOOT file.

The default 'BOOT MENU' setting is used to display the program's help.

Run menu_bas to launch the menu that will display different on-screen help sections.

Program Content

The program includes the following files:

* boot: Boot of the program (use tk2)
* menu_bas: Program startup (without tk2)
* menu_cfg: Setting Menu Parameters
* menu_lst: List of options in the example menu
* menu_scr: Image for menu background
* Infos numbered from 1 to 5 corresponding to the sections of this help and provide a Example of how to use the menu.
* leeme_txt

Also included is a help file called:

* manual_txt

Use

'BOOT MENU' is intended to be freely distributed with its configuration files along with programs. It therefore has no restrictions on use or modification.

The author would appreciate it if you could be quoted in the credits.

Characteristics

Options

The program is capable of displaying a list of applications, highlighting one of them, as well as displaying the information corresponding to each of the entries, allowing you to make your choice using the 'Up' (up), 'Down' (down) and 'Space' (option) keys.

If the configuration file does not exist MENU_CFG, the program will display only a central window with the menu options and their information. This occurs if TK2 is activated or the conf$ value is not specified in the menu_bas program.

The window where the menu and information are displayed can be moved (xw, yw), and resized (w, high). Also the width of the menu can be resized (lon), being the width of the information window inversely proportional to the changes in the width of the menu, that is, if we extend the width of the menu we will reduce the width of the information space. Fortunately we can always change the total width to compensate, having the width of the screen as a limit. You can also swap the order in which the menu and info are displayed.

Entries on the menu are not limited to the height of the window. If the number of menu entries exceeds the character height of the menu window, an indicator of the number of selected options and the total number of menu options is displayed in the window, and the menu options can be scrolled through in the window.

If an option name in 'MENU_LST' contains '_' characters, they will be replaced in the menu by a blank space.

The first line of informational text about each app is automatically highlighted (prime).

We can change all the values regarding the colors of paper (pa_*) and ink (in_*), the shadow of the window, the highlight of the menu, the frame of the windows, etc...

Depending on the configuration of the menu_cfg file, the program can display a wallpaper with a background color (background) and a background image capture on top of it that we will place in the desired screen memory address. The image can be as high as you want (not exceeding the screen height in pixels, i.e. 256), but the width of the image must be the width of the screen (512 or 256, depending on the screen mode), as the data is loaded directly with LBYTES.

In the latter case, a title (xti, yti) is also displayed on one line, a subtitle (xsu, ysu) on two lines and a footnote (xno, yno) on one line, with information about the general content of the disc we are displaying.

The program does NOT require any system extensions, e.g. Toolkit II (TK2) or others, as it is written in standard SuperBASIC, but the use of TK2 allows you to detect if a configuration file exists and makes use of the default values PROG_USE and DATA_USE..., so its use is recommended (1).

Limitations

The program has some limitations:

Requires all files to be in the same location.
The filename without the extension and the menu name must be the same.
The background image must be about the width of the screen and no wider than the screen.

How it works

The program, which has the name BOOT, is loaded using the system's ability to load BOOT files by inserting the media and pressing F1 or F2. From there we called MENU_BAS. We can rename MENU_BAS so that it starts as BOOT.

Set the screen mode (mode) and file location (disp$), load the rest of the variables (VALUES), and draw the screen (DISPLAY).

It then reads the disk's MENU_LST file (LIST) to learn the disk information and applications it should display, and displays them (SHOW). The structure of the MENU_LST file is self-explanatory. See below.

It then gives us the option to select (SELECT) the different menu entries, while showing us the information corresponding to each of them that is found in each _INFO file of each application. E.g. If you have a game called JUEGO_EXE that you want to run, you will have to create a JUEGO_INFO file for the 'BOOT MENU'. This _INFO should be adapted to the size of the program's information window. Your first line will be highlighted in the program. Apart from the _INFO we can attach a _TXT with more specific information if we need more explanations to make the application work, but this _TXT will not be displayed from the program.

Finally, once you've selected the application, all you have to do is press 'Space' to run it (LAUNCH). The 'BOOT MENU' will call the _BOOT of the program you want to run with an LRUN. As with the _INFO in the previous example, we will have to create a JUEGO_BOOT that contains the instructions for loading the application, for example:

```
100 REM Charging Game
110 EXEC_W JUEGO_EXE
```

Once the option is launched, the program ends, unless we have activated the multi value in the MENU_CFG file. In this case, the program allows you to launch as many programs and program instances as you want until you press ESC to exit.

Example

BOOT

The attached BOOT loads the TK2 and launches the program MENU_BAS:

```
 disp$='flp1_' : TK2_EXT : DATA_USE=disp$ : PROG_USE=disp$ : DEST_USE=disp$ : LRUN menu_bas
```

This BOOT is an example of how we can load the values that we will use by default in case we compile MENU_BAS (2). After compilation, we only need to replace LRUN menu_bas with:

```
 EX menu_exe
```

If we're going to run MENU_BAS without compiling, we can even rename MENU_BAS to BOOT and remove the REMarks from the first few lines of MENU_BAS to load the TK2 as outlined in the next section.

MENU_BAS as BOOT

The first lines of the program are MENU_BAS the only ones that you will have to alter if you use MENU_BAS such as BOOT or in any other case. These lines are:

```
 100 REMark programa boot-menu
 110 REMark version 1.4 - jav 2008
 120 :
 130 extension=1 : REMark TK2 loaded S/N
 140 :
 150 IF extension THEN
 160 REMark Clear the following REMarks
 170   REMark si usas 'menu_bas' como BOOT
 180   REMark TK2_EXT
 190   REMark DATA_USE="flp1_"
 200   REMark PROG_USE="flp1_"
 210   disp$=PROGD$
 220 ELSE
 230   disp$="flp1_"  : REMark disp. x def.
 240   ch_fi=4        : REMark canal x def.
 250 END IF
 260 conf$="menu_cfg" : REMark configuraci.
 270 menu$="menu_lst" : REMark lista menu
 280 :
 290 REMark Programa
 300 :
 ...
```

If we activate the extension value (values 0|1, line 130) we will make use of the TK2 and its facilities. The value per defercto is 1, and the rest of the configuration values are intended so that the menu_bas program can be compiled.

In case of using MENU_BAS as BOOT and activating extension, we can remove the REMarks from lines 180 to 200.

If we specify conf$="" (empty string, line 260) the program will take the default values ignoring those of MENU_CFG. Likewise, the default values will be taken if we have the extension variable activated and configured accordingly conf$ but there is no MENU_CFG.

The MENU_LST file and menu$ configuration is a must for the use of the program in this version.

We can replace the conf$ and menu$ values conveniently, and rename the corresponding files.

disp$ will indicate the default directory where to look for the necessary files.

MENU_CFG

We can configure the environment of the menu we are creating using the MENU_CFG file. If the MENU_CFG file does not exist, the program will take the default configuration data.

MENU_CFG contains the values of the variables described in this manual, and is self-explanatory.

Menu Parameters

Through MENU_CFG we can modify the following parameters:

- Display Mode, 512, 256, 8, 4
- Ancho characters in pixels, 6, 12, n

- Character height in pixels, 10, n
- Info Extension: _info
- Boot file extension: _boot
- Does it end when you launch a task? s/n
- Display info window: s/n
- Does it show the background of a color? s/n
- Does it show background image? s/n
- Memory position where to load the image: 131072
- File to be displayed: menu_scr
- Does it display text in the background? (Outside the window) s/n
- Position 'x' and 'y' of the title, in chrs.
- Position 'x' and 'y' of the subtitle, in chrs. (second subtitle line below the first)
- Position 'x' and 'y' of the footnote, in chrs.
- Text 'ink' in the background.
- 'Paper' for the background.
- Position 'x' and 'y' of the menu window, in pixels.
- Width and height of the window in characters. (Takes into account CHRS width.)
- Width of the menu list in chrs.
- Menu on the left or right?
- 'Ink' for the text and 'paper' for the background of the menu window.
- 'Ink' for the text and 'paper' for the background of the selected option.
- Width of the border of the menu window and info window
- Put title in the window frame? s/n
- 'Ink' for the title in the window frame.
- 'Paper' for the background in the frame of the menu windows and info window.
- 'Ink' and 'paper' in the info window.
- 'Ink' and 'paper' for the first line of information.
- Width and height of the shadow in pixels.
- 'Paper' of the shadow.
- Width of the border of the menu and info windows. (margin)
- ASCII code of input keys: up, down, previous, next, redraw, select, exit.

Note: 'Ink' and 'paper' refers to the attributes of the corresponding INK and PAPER instructions.

MENU_LST

Below is the structure of a sample BOOT_INFO file with three applications. The structure and lines with '*****' must be respected up to the line marked as '***** End of data'. Comments can then be added:

```
Program Menu Info
Version - 1 line
1
Description: 4 lines of text
Title Text
Subtitle Text Line 1
Subtitle Text Line 2
Footnote
Total Number of Programs - 1 Line
3
Program Listing - 1 line per program
Application1
Application2
Application3
End of data
What follows the list of programs will be ignored
```

Actually, the lines that are shown with "*****" are ignored, so we can leave them completely blank or put another separator or other text to our liking.

viewing the DIR of the bracket

List of files that will contain the support we use for the previous BOOT_INFO. Files _TXT are not required:

BOOT
MENU_BAS
MENU_LST
MENU_CFG
MENU_SCR
LEEME_TXT
APLICACION1_BOOT
APLICACION1_EXE
APLICACION1_INFO
APLICACION1_TXT
APLICACION2_BOOT
APLICACION2_EXE
APLICACION2_INFO
APLICACION2_TXT
APLICACION3_BOOT
APLICACION3_EXE
APLICACION3_INFO
APLICACION3_TXT

Minimum listed, using MENU_BAS renamed as BOOT (APLICACIONn_* files are not listed in this case):

```
BOOT
MENU_LST
...
```

Calling MENU_EXE (compiled MENU_BAS) from the BOOT (APLICACIONn_* files are not listed in this case):

```
BOOT
MENU_EXE
MENU_LST
...
```

MENU_SCR

With the program, an image is attached that can be used as a header design, and that occupies 1/4 of the screen. By default, the file is called MENU_SCR and is loaded into the 131972 memory position, the first position of the screen memory.

Any other image can be used taking into account the limitations described above.

Running the program

When you run the application, you're shown information about the program using the menu, its title, additional information in the form of a subtitle or hover text, and a text caption.

These options can be disabled and only one option selection window or two windows are displayed, one for selecting options and the other for information regarding the selected menu option.

To move between the different options, use the default keys:

- Up: Up arrow. Move back one position in the options in the list.
- Down: Down arrow. Move up one position in the options on the list.
- Previous: Date left. In a list of options whose size exceeds the height of the window, move back 'n' positions in the list. The 'n' value equals the height measured in menu lines.
- Next: Right date. In a list of options whose size exceeds the height of the window, advance 'n' positions in the list. The 'n' value equals the height measured in menu lines.
- Redraw: F5. Redraw the windows.
- Select: Space. Choose the selected menu option and run the indicated boot.
- Exit: ESC. Exit the app.

Notes

Remember to define the disp$ variable in the BOOT to indicate the path from which the programs will run.

Load the extensions and/or parameters that the program requires for its perfect execution in the _BOOT of each application.

(1) TK2 is already freely distributed and used. It can be found on almost all controllers and expansion cards for QL or alongside emulators. We can load TK2 from disk as the code is fully relocatable.

(2) It will certainly be necessary to review the source code to avoid compiler WARNINGS and make it work faster by substituting, for example, numeric variables with integer variables (variable%) or terminating all IFs with END IF... etc.