QBox

COLLABORATORS			
	TITLE : QBox		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 25, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

QBox

1.1 QBox copyright 1995 T.L.Bullock

Legal texts>

QBox

- Versions v1.14, v1.14μ and v1.14e -

1.2 QBox: Legal texts

<Contents Register>

COPYRIGHT NOTICE

QBox is a copyrighted product that is available to you through the SHAREWARE concept. The full version (vX.XX/vX.XXμ and as indicated in the About window) is NOT public domain (or derivative there of) and

as such any unauthorised copying, hiring, lending, public performance
and broadcasting of this product in its REGISTERED form is strictly
prohibited.
However, the EVALUATION version (vX.XXe and as indicated in the About
window) may be freely distributed as freeware (note: copyright
remains with the author).
This product may only be used under the following terms of license.
TERMS OF LICENSE
All references to "QBox","product", "package", "programs", "software"
and "documentation" refer to the set of files contained within the QBox
archive including any documentation, examples or other files in the
original distribution as produced and published by the author,
T.L.Bullock. This license is for the Commodore Amiga version only.
Any references involving "written permission" refer to documentation
outside of this product's distribution. Any such documents will be
supplied from the author, by the author via "snail" mail, if the author
agrees to any proposals made by parties concerning the use,
distribution, transfer, copying, sublicensing and/or modifications to
the product.
The following terms of licence refer to two different versions: the
evaluation version and the registered shareware version. Unless
otherwise stated conditions refer to both versions. Versions may be
identified by the About window of QBox (clearly stating either
EVALUATION or REGISTERED SHAREWARE therein). Version numbers generally
indicate which form of distribution you have: vX.XX/vX.XXμ for
registered versions, vX.XXe for evaluation versions, so, v1.12e is the
evaluation version of the registered versions v1.12/v1.12μ. Always
check the About window to be sure.
1a. EVALUATION distribution (indicated by vX.XXe):-
The DEMO version may only be distributed in it's original unmodified
form; this archive containing the following unaltered files:-
QBox-vX.XXe : QBox executeable (where vX.XXe is the current
version number e.g. v1.14e)
QBox.guide : This document
Install : Installer script
QBox.readme : Notice
S/ReadMe : Brief note concerning examples
S/Startup.QBox : Example button script
S/StartMenu.QBox : Example menu script

S/QBox/EgButton.QBox : Another example button script
S/QBox/EgMenu .QBox : Another example menu script
Libs/Explode.library : Explodes executeable
Fonts/Basic : Large font
Fonts/Basic/32
Fonts/Basic.font
Fonts/smallfont : Small fallback font
Fonts/smallfont/6
Fonts/smallfont.info
Fonts/Thinpaz : Normal font (10 isnt used)
Fonts/Thinpaz/8
Fonts/Thinpaz/10
Fonts/Thinpaz.font
[All root files above have icons copyright T.L.Bullock]
-If your EVALUATION distribution varies from the above then it is invalid and you should seek out a better supplier (email users may ask for a free EVALUATION distribution direct from the author although the author does not in anyway guarentee reply).
-You may NOT distribute QBox (EVALUATION) without the accompanying unaltered documentation. This documentation MUST contain the copyright notice, terms of license and the disclaimer.
-You may charge a fee to recover distribution costs. This fee for diskette distribution may not exceed the cost of obtaining a public domain diskette from the Fred Fish collection. In any case this value may not exceed 2 sterling (British).
1b. Full REGISTERED SHAREWARE versions (indicated by vX.XX/vX.XXμ):-
-You may NOT copy, modify, distribute, sublicense or transfer the REGISTERED version of this product without the witnessed written permission of the author. Any attempt to do so will be seen as an infringement of copyright and may lead to prosecution.
-No person(s) or business(es) other than the author are authorised to accept any registration fees or distribution fees in any form for the SHAREWARE versions of QBox. Commercial licenses concerning the distribution of the SHAREWARE versions are subject to negotiation with the author (see Contact for address}. The archive should contain:-
QBox-vX.XX : QBox executeable (where vX.XX is the current version number e.g. v1.14)
QBox-vX.XXμ : Micro (mini) version
QBox.guide : This document

QBoxQRef.gif : Gif image for identifying button functions
Install : Installer script
QBox.readme : Addendum and notice
S/ReadMe : Brief note concerning examples
S/Startup.QBox : Example button script
S/StartMenu.QBox : Example menu script
S/QBox/EgButton.QBox : Another example button script
S/QBox/EgMenu .QBox : Another example menu script
Libs/Explode.library : Explodes executeable
Fonts/Basic : Large font
Fonts/Basic/32
Fonts/Basic.font
Fonts/smallfont : Small fallback font
Fonts/smallfont/6
Fonts/smallfont.info
Fonts/Thinpaz : Normal font (10 isnt used)
Fonts/Thinpaz/8
Fonts/Thinpaz/10
Fonts/Thinpaz.font
[All root files above have icons copyright T.L.Bullock]

2. Coverdisks and Compilations

This product may not be distributed on magazine coverdisks or as part of a compilation disk in any form without written permission of the author. See [1b] for further restrictions.

3. Hacking etc.

Under no circumstances may any copyright notices, license notice, disclaimer or program data be altered, hacked or otherwise changed.

4. Distribution

You may not copy, modify, distribute, sublicense or transfer this product except as expressly provided under this license. Any attempt to otherwise copy, modify, distribute, sublicense or transfer this product is void, and will terminate your rights to use this product.

5. Licensing

Any parties recieving a valid copy of this product will be be subject to the same terms of license as stated here unless the author gives witnessed written permission to the contrary. In the event of such permission the terms stated therein are directed explicitely at said parties. Any receiver of a distribution made by said parties are bound by this license only. No subsequent restrictions may be imposed and/or

amended to those granted herein.

6. Acceptance of Terms

By using, copying and/or distributing this product you indicate your acceptance of this license to do so and all accompanying terms and conditions.

7. Reverse Engineering

You may NOT decompile, dissassemble, re-source or otherwise reverse engineer any of the programs that make up this product.

8. Product Use

You may not use any part of this product in part or in whole in other program(s) without the written permission of the author, with the exception of this license notice, which may be altered appropriately for your own product. This is done at the users own risk.

9. Cessation of Distribution

You agree to cease distributing this product (programs and data) if requested to do so by the author, even if the author has granted permission previously.

10. Withdrawal of Product

The author reserves the right to withdraw the product from the market place without warning. The author will only make such alterations where a new version and/or the economic climate form a reason to do so.

11. Price Changes

The author reserves the right to alter subscription charges for newer versions without advance any notice. Orders made to the value of a previous version will only be entitled to that previous version.

DISCLAIMER

THIS PRODUCT COMES WITH NO WARRANTY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED THE COPYRIGHT HOLDER AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY OF PERFORMANCE OF THIS PRODUCT IS WITH YOU, THE PRODUCT USER. SHOULD THE PRODUCT PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR OTHER PARTY WHO MAY REDISTRIBUTE THE PRODUCT AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT

OF THE USE OR INABILITY TO USE THE PRODUCT (INCLUDING BUT NOT LIMITED
TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED
BY YOU OR THIRD PARTIES OR A FAILURE OF THE PRODUCT TO OPERATE WITH
ANY OTHER PRODUCTS). EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN
ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
EVERY EFFORT HAS BEEN MADE TO ENSURE THAT THIS PRODUCT IS VIRUS FREE.
HOWEVER, THE AUTHOR TAKES NO RESPONSIBILITY FOR THE PRESENCE OF ANY
VIRUSES THAT MAY HAVE INFECTED YOUR DISTRIBUTION.
DISTRIBUTION VERSION NOTES
You may still be confused as to which version of QBox you are using.
As of v1.12e all evaluation versions are denoted by vX.XXe. That is
v1.13e, v1.14e, v2.00e all denote evaluation versions. The word
EVALUATION should be present in the About window too.
Note: No evaluation version was available prior to v1.12e.
Registered versions are denoted by vX.XX and vX.XXμ (the vX.XXμ a cut
down version of vX.XX for general day to day use, where as vX.XX
contains extras such as the preference editors). So, v1.13 and v1.13μ
are both REGISTERED versions, so too are v1.14 and v1.14μ, v2.00 and
v2.00μ. The words REGISTERED and SHAREWARE are present in both vX.XX
and vX.XXμ About windows.

1.3 QBox: Registration

<Legal Texts Introduction>
Registration
Sick of the restrictions of the Evaluation version? Sick of only being able
to have 4x4 buttons? Sick of being restricted to I/O buffers of upto 1024?
Sick of having to manually edit button and menu scripts? Sick of only being
able to add 3 menu items? Sick of that annoying About requester? Well,
theres a simple solution - REGISTER! Registering for QBox is a cheap way of
saving lots of frustrated hair pulling. Go on, treat yourself...
What do you get?
Registered users will receive a disk containing the latest release (or the
last release your fees cover (see Terms of license section 11) of QBox. You
will also receive the most uptodate release of TLB-Tools covering sorting,
cross-referencing, file subtraction, basic file processing etc. Further
more, special discounts may be extended to registered users for future
updates and other shareware titles by the same author.

Registered versions have all those annoying forced About windows, I/O, matrix and menu list restrictions removed. Not satisfied? Well, the registered distribution contains two versions of QBox: a full version with preference editors and a cut down version for day to day use. It also comes with a quick reference image to help you in learning QBox's numerous editor buttons.

How to Register - private (Read carefully)

In order to receive the latest REGISTERED SHAREWARE version you must first send a registration fee to the value of £2 (UK sterling). An additional sum must be paid to cover postage:

UK Residents

In addition, people recieving a copy on mainland Britain should add £1.50 (UK sterling) for each copy of QBox ordered to cover postage and packaging. The total fee (£3.50 UK sterling) should be mailed to the author at the below address in the form of a postal order made out to "T.L.Bullock" in the payee line.

Non-UK Residents

People ordering from outside of mainland Britain should add £3.00 (UK sterling) to cover post and packaging. The total fee (£5.00 UK sterling) should be mailed to me in British legal tender (ie. a British five pound note) sandwiched in a greetings card to disguise the envelopes contents.

NOTE: I disclaim any responsibility for the loss of any moneys lost in the postal system. By mailing me cash you take full responsibility until it reaches me personally. I will only mail a registered version of QBox when the correct moneys have reach me personally.

Please send the correct registration fees and the registration form (template and examples elsewhere in this section) to:-

Tony Bullock
c/o 74A Shilton Road,
Carterton,
Oxfordshire,
OX18 1EL,
England (Europe).

Mark the top left corner of the envelope with "QB".

Registration Form (template)

Please include the following details (print out the following few lines if you like):-

Product: QBox Current Version: v__.____

Name: _____

Address: _____

_____

_____

_____

Phone: _____

Where did you obtain your QBox EVALUATION version? _____

_____

_____

Fees Paid: Registration £2.00

Postage (Mainland UK £1.50, Overseas £3.00) +£_.__

Example form:-

_____

| |

| Product: QBox Current Version: v1.14e |

| |

| Name: Fred Bloggs (IRC nick: Bloggy) |

| Address: 1176 Mane Street, |

| Yawk, |

| Yawkshyre, |

| England. |

| Phone: +44 01913 013049 |

| |

| Where did you obtain you QBox EVALUATION version? |

| src.doc.ic.ac.uk:pub/aminet/util/wb/QBox-v1.14e.lha |

| |

| Fees Paid: Registration £2.00 |

| Postage +£1.50 |

|_____|

Please allow anything up to four weeks for despatch. You will usually receive your registered version ASAP, but in the event of a new version nearing completion I may delay until the new version is completed (unless you request that I send you a version immediately).

NOTE: The prices shown above are correct at the time of writing this doc (27th Nov 1995) and are subject to variation. The author will only make such alterations where a new version and/or the economic climate form a reason to do so. If in doubt of the current price you may write for a quote to the above address or email me on e0190162@brookes.ac.uk.

How to Register - commercial ventures

This product can be registered to a group/company (subject to the authors disgression) for a set fee negotiated with the author. Please contact the author for further details.

1.4 QBox: Introduction

<Register Installation>
Introduction
QBox is a SHAREWARE utility that is designed to aid you, the user, in launching applications and CLI programs.
QBox, can be used to save typing in CLI by enabling you to specify button/menu text and associated command line calls (and even scripts) simply, with no fuss or hassle.
Many utilities have attempted (some even succeeding) to fill this role, however, complicated preference systems, command data restrictions and extortionate registration costs have dogged most, if not all, such systems. The QBox project is an attempt to fill some of these gaps.
#
@ @
---oOO-(_)-OOo---

1.5 QBox: Installation

<Introduction Usage>
Installation
Requirements
Before I list the requirements I should mention I have only tested this program on a vanilla A1200 (okay its got an 80mb HD and an external XL drive - but no extra ram, fpu, mmu, 030 (gimme), 040 (yes please), 060 (donations welcome)) and a CD32.
Required:
An Amiga - PC, Mac, Oric, BBC B and other similarly (un)sophisticated users need not apply.
OS3.0+ - OS3.0 and OS3.1 tested. Might work with OS2.04+.
GadToolsBox.library - v38.45+ (some earlier versions might work)
ASL.library - v39.4+ (some earlier versions might work)
DiskFont.library - v39.3+ (some earlier versions might work)
Recommended:
A harddisk - Not really necessary but it speeds things up and is what QBox was really designed for.
Saying that floppy user shouldn’t be put off.
Installing QBox
Simply click on the Install-QBox icon and enter data as prompted.

Alternatively, type the following:-
copy QBox Sys:wbstartup/QBox
copy QBox.info Sys:wbstartup/QBox.info
copy QBox.guide Locale:help/english/QBox.guide
copy fonts fonts: all clone
The various libraries required (see above) are present on your
Workbench 3.x disk.
The author claims no copyright of the freeware or PD fonts used in
QBox.
Things you should know
QBox will, when started without a BUTTON parameter (see Usage) , attempt
to locate a preference file called "S:Startup.QBox". If this file does not
exist then a default matrix is used.
At startup QBox will also attempt to load a menu preference file called
"S:StartMenu.QBox" if MENU parameter is not specified.
A temporary file called "RAM:QBoxEXEC.temp" is created whenever the \S
option is used in a script. This is removed after QBox
has finished running the QBox script.
Help calls up Sys:utilities/MultiView and loads Locale:help/english/QBox.Guide.
Both may be changed via ToolTypes.

1.6 QBox: Usage

<Installation Main Window>
Usage
QBox can be started from either workbench or shell. Either way you will
open a "Main" window filled with a number of buttons (default is 5).
Quick Start (links only available to REGISTERED users)
Show QRef Image (in guide)
Show QRef Image (PPShow 640x512x4)
General Usage
Main Window « Start here then read as required
Pull-down Menu
Button Prefs [Not evaluation version]
User Menu Prefs [Not evaluation version]
Misc Prefs [Not evaluation version]
Tool Types
File Formats « Evaluation users need to know this
Scripts « All users need to know this

Shell usage
Synopsis: QBOX [[BUTTONFILE=]<file>] [[MENUFILE=]<file>]
[[TIMEOUT=]<n>] [[DEFFILE=]<file>]
Template: BUTTONFILE,MENUFILE,TIMEOUT/K/N,DEFFILE
Options :
BUTTONFILE Path and filename of QBox button prefs file
MENUFILE Path and filename of QBox menu prefs file
TIMEOUT/K/N Period QBox runs with no interaction
DEFFILE Path and filename of amigados script if QBox times out
Example :
QBOX S:BootStart.QBox S:BootMenu.QBox 20 S:Startup-Sequence
means start QBox and load S:BootStart.QBox into the button matrix and
S:BootMenu.QBox into the user menu list. If the user fails to interact
with QBox for 20 seconds, QBox will launch S:Startup-Sequence and
automatically quit (ideal for choosing multiple startups)
Defaults (unless otherwise defined in icon ToolTypes):
BUTTONFILE S:Startup.QBox
MENUFILE S:StartMenu.QBox
TIMEOUT/K/N Infinite (ie. TIMEOUT<1)
DEFFILE Null
Note :
DEFFILE Only works in conjuction with TIMEOUT/K/N
The DEFFILE script should be a standard AmigaDos script. However, in order
to control how this script is run you can prepend the file with ;\ followed
by a QBox command (RNXH) to the first line. You may not use comands B, M
and S. For more on QBox commands see script.

1.7 QBox: Usage / Main Window

<Usage Button>
Main Window
QBox v1.12 ©1995 T.L.Bullock window
Prefs
Button Button
Button Button
This window is where the day to day use of QBox will occur once the
user has configured the preferences to their liking. For the purposes
of documentation we shall call this the Main Window although
any user-defined title may be used in the title bar.

The button marked Prefs opens all three preference editors. If all three
are already open then they are all closed. This button is only available
in the REGISTERED version of QBox.
Registered users see Button Prefs
User Menu Prefs
Misc Prefs

1.8 QBox: Usage / Main Window / Button

<Main Window Button Prefs>
Buttons
QBox supplies numerous user-definable buttons ("prefs" is NOT user-definable).
The number of these buttons is defined using Dim X and Dim Y in the
Button Prefs window.
Each button consists of an appropriate button name with an associated
command script. These are defined in the Button Prefs window in the
text entry boxes labelled Button and Command
respectively.
Activating a button can be done by pressing the left mouse button when the
mouse pointer is over the desired button OR by pressing the defined
keyboard shortcut (indicated by the underlined letter in the button
text).
Activating on a button will result in one of two actions, depending on
whether or not the Button Prefs window is visible:-
a) With the Button Prefs window hidden (default) the command script for
the button selected will be executed.
b) When the Button Prefs window is visible the button name and command
details will appear in the Button Preference window for editing purposes.
The current command script will not be executed in this mode of
operation.

1.9 QBox: Usage / Menu

<Button New (Button)>
Pull-Down Menu
Project User
New* UDM 1..N
» Button/Misc*
» User Menu*

Load

» Button/Misc

» User Menu

Save*

» Button/Misc*

» User Menu*

SaveAs*

» Button/Misc*

» User Menu*

Delete*

Prefs*

» Button*

» Menu*

» Misc*

» Show About*

» Show Prefs*

» Case Buttons*

Hide

Help

About

Quit

* Indicates option unavalable to EVALUATION users.

Registered users will also notice that the * features are not present in the REGISTERED vX.XXμ version but are available in the full vX.XX also included in your distribution. The vX.XXμ is designed to run QBox data, whilst vX.XX is designed to edit said data.

1.10 QBox: Usage / Menu & Button Prefs / New (Button)

<Menu New (UDM Prefs)>

New (Button)

Hotkey: Right Amiga + C

This menu item / button prefs gadget clears the current matrix leaving the dimensions untouched. You will NOT be asked to verify your actions so take care!

1.11 QBox: Usage / Menu & Menu Prefs / New (UDM Prefs)

<New (Button) Load (Button)>

New (UDM Prefs)

Hotkey: Right Amiga + V

This menu item / menu prefs gadget clears the current menu list leaving the a single hidden menu item. This item can be given a name and thereby be brought into view. You will NOT be asked to verify your actions so take care!

1.12 QBox: Usage / Menu & Button Prefs / Load (Button)

<New (UDM Prefs) Load (UDM Prefs)>

Load (Button)

Hotkey: Right Amiga + L

This button / menu item brings up an ASL file requester allowing you to select a button preference file (created and saved previously with QBox). When you have selected a suitable file, the file will be loaded into the button matrix (replacing all existing definitions).

A simple file format check is carried out on any selected file to ensure that menu prefs files can not be loaded into the button matrix. You will be alerted if you attempt to load a menu prefs file.

1.13 QBox: Usage / Menu & Menu Prefs / Load (UDM Prefs)

<Load (Button) Save (Button)>

Load (UDM Prefs)

Hotkey: Right Amiga + O

This button / menu item brings up an ASL file requester allowing you to select a menu preference file (created and saved previously with QBox). Once a suitable file has been selected, said file will be loaded into the UDM menu list (replacing all existing definitions).

Basic file format checking is carried out to ensure that button prefs files can not be loaded into the menu list. You will be alerted to any attempts to load a button prefs file.

1.14 QBox: Usage / Menu / Save (Button)

<Load (UDM Prefs) Save (UDM Prefs)>

Save (Button)

Hotkey: Right Amiga + W

This menu item enables you, the user, to write the current button matrix over the last loaded button prefs file. There is no file requester and no confirmation, so only select this if you mean it!

1.15 QBox: Usage / Menu / Save (UDM Prefs)

<Save (Button) Save As (Button)>

Save (UDM Prefs)

Hotkey: Right Amiga + E

This menu item enables you to write the current UDM menu list over the last loaded menu prefs file. There is no file requester and no confirmation.

1.16 QBox: Usage / (Menu / Save As) & (Button Prefs / Save) (Button)

<Save (UDM Prefs) Save As (UDM Prefs)>

Save As (Button)

Hotkey: Right Amiga + S

This menu item / button prefs gadget allows you to save the current button matrix to a button prefs file. Upon selecting this option you will be presented with an ASL file requester. Simply select the required file (or enter a new filename) and click on the button labelled "OK". There is no confirmation - the matrix will be saved.

1.17 QBox: Usage / (Menu / Save As) & (Button Prefs / Save) (UDM Prefs)

<Save As (Button) Delete>

Save As (UDM Prefs)

Hotkey: Right Amiga + D

This menu item / menu prefs gadget allows you to save the current UDM menu list to a menu prefs file. Upon selecting this option you will be presented with an ASL file requester. Select the required file (or enter a new filename) and click on the button labelled "OK". There is no confirmation.

1.18 QBox: Usage / Menu / Delete

<Save As Show About>
Delete
This menu item enables you to delete any (unprotected) AmigaDOS file. On selection you will be requested to choose a file through an ASL file requester. Simply select the file for deletion and click on "OK". You will then be asked if you are sure you want to delete the selected file. If you are then click on "Do It" and file will be deleted.
You may abort this process by clicking on the "Cancel" button in the ASL requester or the "Cancel" button in the "Are you sure?" requester.

1.19 QBox: Usage / Menu / Show About

<Delete Show Prefs>
Show About
Hotkey: Right Amiga + A
This menu toggle enables you to prevent the About text popping up after loading a given preference file.
This toggle will only affect preference files saved with the toggle set into the "blank" position (ie. no tick next to "Show About"). It will not affect all preference file loading, just those saved with Show About disabled. The default setting is in the tick position.
To use this feature you must set the toggle to a tick (show about) or blank (hide about) and save the preference file with either Save or Save As.

1.20 QBox: Usage / Menu / Show Prefs

<Show About Cased Buttons>
Show Prefs
Hotkey: Right Amiga + P
This menu toggle enables you to define whether or not the Prefs button is visible (when a tick is next to "Show Prefs" menu item) or hidden (no tick) after loading.
The toggle (like the Show About one) only affects preference files saved with the toggle set to the "blank" position (the default is a tick). Again this is specific to each preference file, not to all load operations.
Unlike Show About, you action will become immediately apparent in the Main window as the Prefs button appears and disappears.
To use this feature you must set the toggle to a tick (show prefs) or blank (hide prefs) and save the preference file with either Save or Save As.

1.21 Qbox: Usage / Menu / Cased Buttons

<Show Prefs Hide>

Cased Buttons

Hotkey: Right Amiga + K

This menu toggle allows you to define whether or not the current button matrix keyboard short-cuts are case sensitive or not. Case sensitivity defaults to on. If you wish to change this then simply toggle this menu option so that no tick is displayed next to the menu text.

In summary, a tick on this menu item means keyboard short-cuts for each button are case sensitive (eg. _a is not the same as _A). No tick means that case sensitivity is off (eg. _a is the same as _A).

You will notice that in the screen title bar there is, to the right, an area dedicated to informing you of the case sensitivity for the current button matrix (either <Case>/<No Case>). This is particularly useful for determining case sensitivity for the current matrix in a multilayered matrix systems (where buttons are linked to other matrix definitions via ;\B).

This setting is saved when the button matrix is saved and is loaded whenever a button matrix is loaded, either manualy or from a button/menu script.

1.22 QBox: Usage / Menu / Hide

<Cased Buttons Help>

Hide

Hotkey: Ctrl + Left Amiga + Z or current popkey$

This menu option allows you to hide QBox from view until such time as you require to see it again. This is possible because QBox is a commodity (see Amiga Use Guide).

To make QBox reappear you should select "Show Interface" in the Exchange program supplied with workbench (tools/commodities drawer) or press the appropriate hotkey (default to Ctrl LCommand Z, but may be altered via the ToolTypes).

1.23 QBox: Usage / Menu / Help

<Hide About>

Help

Hotkey: Right Amiga + H

This menu option will attempt to launch a MultiView session that will load
the QBox.guide. QBox assumes that MultiView is in one of the directories
defined with Path. If this is not the case then edit your user startup,
adding the following line before any call to QBox:

Path Dh0:utilities ADD

(Dh0:utilities is where MultiView is on Workbench 3.x distributions.)

AmigaGuide users should add the following line instead:

Alias MultiView Dh0:c/AmigaGuide

(You may have to adjust the AmigaGuide pathname to suit your installation.)

The QBox.guide must be present as:

Locale:help/english/QBox.guide

You may use the SHOWGUIDE and GUIDE Tool Types to alter the
viewer and guide path and filenames.

1.24 QBox: Usage / Menu & Button Prefs / About

<Help Quit>

About

Hotkey: Right Amiga + ?

This button / menu item displays a brief notice window. Unlike other windows
in QBox, this window has no close gadget. To close the window simply press
any mouse button or key.

As of v1.12 the distribution version is indicated so that you can identify
whether or not your version is REGISTERED or a simple EVALUATION version.

REGISTERED version Abouts also display who the program is registered to.

The full conditions of use are made clear in the Legal Texts section of
this document.

1.25 QBox: Usage / Quit & Menu / Quit

<About User Definable Menus>

Quit

HotKey: Right Amiga + Q

Fairly obvious this one. Select this menu option (or the close gadget in

the main window) and a requester will appear asking you whether you really want to "quit", "hide" or resume (labelled "Cancel") QBox. Clicking on "Hide" has the same effect as the Hide menu option.
It is also possible to define a QBox button (see Prefs) to quit QBox.
For this read the Script section.
NOTE: QBox attempts to clean up after itself, but as my version of Blitz2 leaves 40k when using GadTools you will lose some memory. As soon as I have time I will sort this out (apparently BUM6 has a fix for this).

1.26 QBox: Usage / Menu / User Definable Menus

<Quit Button Prefs>
User Definable Menus
As of version 1.06 QBox has supported User Definable Menus (UDMs). UDM item(s) appear beneath the menu title "User".
UDM items operate in a similar way to buttons in that they both have a name with an underlying script. In fact, menu scripts utilise all of the features found in button scripts. A full list of these features can be found in then scripts section.
UDMs are editted via an in-built Menu Editor.

1.27 QBox: Usage / Button Prefs

<User Definable Menus App Window>
Button Prefs
Hotkey: Right Amiga + B
QBox Button Prefs window
Button Button Name
Command Command script
Copy Exch Clr Undo Rvrt X X Y Y
New Load Save Iprt Eprt DimX Dim X DimY Dim Y
This window is used for defining the matrix size and button definitions of the Main window.
This window is an App Window. That is, if an workbench icon is dropped on to this window various attributes will be filled in. Click on App Window for more.

1.28 QBox: Usage / Button Prefs / App Window (Button)

<Button Prefs Button Name>

App Window (Button)

The Button Prefs window is an App Window. That is, a window that, when a workbench icon is "dropped" on it, takes various information from the icon. In this case, QBox enters the name of the file (to which the icon belongs) and puts said name in the Button Name edit area of the Button Prefs window. At the same time it takes the full pathname and filename of the file and places it into the Button Command edit area of the same window.

QBox also recognises multiple icon selections (see Amiga User Guide). The button name will take the name of the first file processed. The button script will take all of the full filenames separating each with a semicolon (;). The order of the button script depends solely on the order in which workbench passes the icons to QBox.

1.29 QBox: Usage / Button Prefs / Button Name

<App Window Command Script (Button)>

Button Name

This text entry gadget enables you to alter the name of a chosen button in the Main window. That is, the visible text part of a button.

Choosing a Button

To choose a button open the Button Prefs window and then click on the button you wish to alter (in the Main window). You should see the details for the selected button appear in the Button Prefs window.

You can click in the Main window and press the user-defined keyboard short cut for a desired button.

Altering a Button

Now that the button has been chosen, click in the text entry gadget labelled "Button" (in the Button Prefs window). Alter the text using the keyboard (cursor keys move the cursor within the gadget). When you have finished making alterations in the text entry gadget press the Return key (the largest grey key). You should now see that the button in the Main window is altered to reflect the new name text you have just entered.

Keyboard shortcuts can be made by placing an underscore before the letter deigned to be the shortcut. Eg.

Q_Box keyboard shortcut will be B during normal use

QBo_x keyboard shortcut will be x during normal use
Note that keyboard shortcuts in the main window are case sensitive (other windows are not) and that each shortcut should be unique. If two buttons have the same shortcut then only the first occurence will be executed when in normal use (even though both buttons will have the underline under the same letter).
This sounds complicated but is very intuitive (no pun intended).

1.30 QBox: Usage / Button Prefs / Command Script (Button)

<Button Name Scripts>
Command Script (Button)
This text entry gadget (labelled "Command") allows an AmigaDOS / QBox script to be associated with a given button (in the Main window). That is, the invisible functional part of a button item.
Choosing and altering a button is done in a similar way to altering the Button Name. However, any changes are not immediately apparent (the button name will not change). You can, however, verify any alterations by re-choosing the button after editing if you need to.
See Scripts for syntax.

1.31 QBox: Usage / Button Prefs & Menu Prefs / Scripts

<Command Script (Button) Copy Button>
Scripts
A QBox button/menu script is usually defined as a string of one or more AmigaDOS commands, with each command separated by a single semicolon (;).
Eg1:
Echo "Line 1";Echo "Line 2";Echo "Line 3"
As this example stands the script will be run to a new shell window. This window will remain unclosed locking-out QBox (with any interaction being buffered) until you type "EndShell" in the new shell window. To prevent this lock-out, QBox supplies a number of script commands (or run tags). These commands should be appended to the right of the script after ";\", thus:
cmd1;cmd2;cmd3;cmdN;\RNX
buttonprefsfile;\B
menuprefsfile;\M
where R means return from CON (ie EndShell)

" N " run to Nil: (useful for hiding CON:)

" S " force button script to an temporary AmigaDos version in RAM

" B " all text before ;\ is treated as QBox button prefs filename

" M " all text before ;\ is treated as QBox menu prefs filename

" X " exit from QBox as soon after launch as possible

" H " wait for mouse or joystick button depression after execution

So, to remedy the Eg1:

Eg2:

Echo "Line 1";Echo "Line 2";Echo "Line 3";Wait;\R

Launching an application (such as a second QBox session) can be done thus:

Eg3:

QBox RAM:startup.own.QBox;\RN

Overides

Notice how \R and \N are used together as \RN or \NR (order is not important). \N will not open a shell window. \B and \M override all other QBox commands. So,

script;\RSNXB -> button_prefs_filename;\B

script;\RSNXM -> menu_prefs_filename;\M

script;\RSNX -> script;\RSNX

Script-Only Commands

\S is used to get around a nasty (although unavoidable) problem in the way that AmigaDOS executes QBox scripts. A QBox script is NOT the same as an AmigaDOS script. AmigaDOS contains a number of script-only commands that can not normally be run be typed in at a shell (such as .key, failat, lab, skip etc). In normal operation QBox simply submits a batch job of AmigaDOS commands, NOT a true script hence script-only commands fall over. \S forces QBox to convert a QBox script into a temporary AmigaDos script situated in RAM:QBox.tempexe. This file is then executed and deleted. As a true AmigaDos script all script-only commands can be used.

In short, use \S if a QBox script seems to fail for no apparent reason.

So, say we want...

;AmigaDOS script

.key message

failat 21

Echo "<message>"

Wait

;End script

...you should have the following QBox script:

;Button script

.key message;failat21;Echo "<message>";Wait;\SR

;End script

Note: .key will still remain redundant as there is no method (currently) available to define a CLI argument within QBox. .def should always be used for this reason. Most other script-only commands should work properly with \S.

Loading QBox Button scripts from a Button/User-Menu

\B will force all previous script to be treated as the path and filename of a QBox button prefs file and ignore other QBox commands. If the preference file exists, the button matrix will be reconfigured to the data outlined in the preference file. In this way you can define multilayered QBox button matrices without launching several QBox sessions. A limiting factor to this is that you can only have one matrix visible at a time. If you want to be able to move back to the previous script you will need to define a menu/button that loads the appropriate preferences with a \B (there is not internal mechanism - yet).

Loading QBox Menu scripts from a Button/User-Menu

\M is the menu equivalent of \B and operates in much the same way (only it loads menu scripts rather than button scripts).

Loading QBox Button and Menu scripts from one button/menu

\M and \B may be combined into a special case as \MB or \BM. The order of the merge determines which part of the script is used for each part. So,

buttonfile;menufile;\BM -> buttonfile;\B & menufile;\M

menufile;buttonfile;\MB -> menufile;\M & buttonfile;\B

buttonfile;\BM -> buttonfile;\B

menufile;\MB -> menufile;\M

buttonfile;menufile;\B -> buttonfile;\B

menufile;buttonfile;\M -> menufile;\M

Detaching commands within a script

Programs in a QBox script are normally executed as each command is completed in turn, sequentially one after another (left to right).

Some programs halt this process until said program has completed or are terminated, with any further commands running afterwards.

The result is that QBox may lock up (buffering any user interaction) until the button script has completed.

To move around this problem you can prepended any commands that may cause lockout with "Run >NIL: ". This AmigaDos call forces the command

to run in the background, detaching itself from QBox. "Run " can also
be used to achieve this, but may open a shell window in the process.
In either case, be aware that detaching a command will mean that the
next command in the button script may be executed before the detached
command has finished. This means that sequential behaviour may be upset.
You can get around this by building a standard AmigaDos script externally
to QBox (saving it somewhere) and call said script with
Execute >NIL: scriptname;rest-of-script;\qbox-cmds
Demonstration:
(you will need to create a text file called "List" before trying this)
Eg1: Without external script:
;Button script
Run >NIL: C:Sort List List;More List;Echo "Check if List is sorted";\R
;End script
Eg2: With external script:
;External script: S:SequentialScript1
C:Sort List List
More List
;End script
;Button script
Execute >NIL: S:SequentialScript1;Echo "Script running in background";\R
;End script
The first script should result in an unsorted List being displayed. This is
because "More List" is executed as soon as "C:Sort List List" is launched
(it does not wait for Sort to complete it’s task). The second however,
will display a sorted List. This is because although the external script
is ran in the background, the contents of said script will be run in order.
You could achieve the same effect by making a self-writing button script
that creates a temporary external script, runs it and removes it, thus:-
;Button script
Echo >RAM:temp "C:Sort List List";Echo >>RAM:temp "More List"; Execute
>NIL: RAM:temp;Echo "Script running in background";Delete Ram:temp;\R
;End script
Auto-Quitting QBox
\X is used to quit QBox as soon as the script has released complete control
from QBox (see \S above). The interface will be hidden as soon as the
script has been launched thereby saving on memory until such point as QBox
can close itself properly. During this hide-quit stage the only trace of
QBox is to be found in the CX Exchange list (and the consumed memory).

Pausing Output
\H is used to hold any window output on screen until such point as either
a joystick fire button or any mouse buttons are pressed. This is of most
use when debugging QBox button scripts, saving you having to add the WAIT
AmigaDOS command into scripts.
Notice that pausing will take place whether or not any window was opened
during script execution. For this reason it is adviseable that \H should
only be used when a Shell window can be guarenteed to appear or when
debugging to ensure easy user interaction. Also avoid using \H in finished
button script lines that use \N or >NIL: redirection (as window output
should be supressed).
Embedding Semicolons
As you will have noticed, QBox uses semicolons to seperate script
commands. Most of the time this causes no problems, however, what if we
want to display text with a semicolon in it? Up until v1.13 you would
now be chewing you nails. Fortunately, v1.13 and above interpret script
data such that semicolons can be presevered in quotes, thus:-
Echo "Eg; Text";Echo "Next Command";\RH
The first quote of a pair of quotes indicates text that is not parsed
(ie. semicolons are not command seperators here), and the second quote
indicates where parsing resumes (ie. where semicolons become seperators).
If an odd number of quotes are present then the script is virtually
appended with another quote. So,
Echo "Eg; Text;Echo Next Command;\RH
Becomes:-
Echo "Eg; Text;Echo Next Command";\RH
Be careful!!!

1.32 QBox: Usage / Button Prefs / Copy Button

<Scripts Exchange Button>
Copy Button
Clicking on this button will cause the screen title to change informing
you to select a button or press both mousebuttons to abort the copy
function.
Clicking on a button in the Main window will result in the current
button (whose values are in the Button Prefs window) being copied
to the one clicked on.
This can be particularly useful for "modelling" buttons. These "models"
can then be adjusted to suit individual requirements.

1.33 QBox: Usage / Button Prefs / Exchange Button

<Copy Button Clear Button>

Exchange Button

This button operates not unlike Copy Button.

Clicking on this button will cause the screen title to change informing you to select a button or press both mousebuttons to abort the exchange function.

Clicking on a button in the Main window will result in the current button (whose values are in the Button Prefs window) being exchanged with the one clicked on. In effect, swapping all button attributes - ideal for rearranging QBox button layout.

Tip: When turning a horizontal button arrangement into a vertical one make keep DimX the same and adjust DimY. Now use Exch to swap newly created blank buttons with the ones that need moving. When you have finished alter DimX to the new (narrower) value. This way you don't have to re-enter button values. See below:-

t t+1 t+2

Original > Transition > New Layout

DimX = 5 DimX = 5 DimX = 2

DimY = 2 DimY = 5 DimY = 5

A B C D E A B C D E A F

F G H I J F G H I J B G

- - - - - C H

- - - - - D I

- - - - - E J

1.34 QBox: Usage / Button Prefs / Clear Button

<Exchange Button Undo (Button)>

Clear Button

This allows you to clear the button definition for the currently selected one (the one whose text is visible in the button prefs window).

That is, button and it's associated button script are set to null (i.e. blanked).

That's all there is to it really.

1.35 QBox: Usage / Button Prefs / Undo (Button)

<Clear Button Revert (Button)>

Undo Button

This button in the Button Prefs window enables your last button edit action to be undone. That is, if you edit a buttons attributes (name or command) and decide that the new button attributes are undesirable, you may, by clicking on this button, force the button attributes to revert to the values preset prior to the edit.

This operation will Undo the following edit events:-

App Window

Button Name

Button Command

Clear

Copy

Exchange

Import

Export

Be aware

~~~~~~~~

Undo's will only affect the very last action in the Button Prefs window. If you type some unwanted text as a button name and press the Return key, then click back in the button name edit area and press the return key again, you will not be able to revert back to the original name.

The table below is an attempt to convey what`s going on. t+x is used to show subsequent edits/undos. The initial button setting is "Original" (t), which is then changed to "Bad Edit" (t+1). The user then hits return in the edit field a second time thereby forcing the new and old value to be the same (t+2). Finally the user then tries to undo "Bad Edit" back to "Original" but finds that they can't.

|| t | t+1 | t+2 | t+3

|| Initial | Edit | 2nd Return | Undo

----------++----------+----------+----------+--------------------

current name || Original | Bad Edit | Bad Edit | Bad Edit _ Swapped

undone || <empty1> | Original | Bad Edit | Bad Edit /

forgotten || <empty2> | <empty1> | Original | Original
~~~~~~~~

1.36 QBox: Usage / Button Prefs / Revert (Button)

<Undo (Button) X>

Revert (Button)

This button can be seen as either a heavy handed Undo or a Reload option.
Both descriptions are applicable here.
Basically, Revert (Rvrt) loads the last loaded/saved button script thereby undoing any changes and reverting the button matrix back to its original form.
There is no verification (if there were there would be no benefit over simply using load) so be careful.

1.37 QBox: Usage / Button Prefs / X

<Revert (Button) Y>

X

This numeric entry box allows you to see the X-ordinate of the button you are editing (in terms of it's horizontal position in the button matrix).
You may also enter a positive integer greater than 0 and less than the matrix width (measured in number of buttons) in order to edit the button at this position. The vertical position can be adjusted with Y
No data is altered during this process. It is simple a way of selecting a button for editing in the Button Prefs window.

1.38 QBox: Usage / Button Prefs / Y

<X Import (Button)>

Y

This numeric entry box allows you to see the Y-ordinate of the button you are editing (in terms of it's vertical position in the button matrix).
You may also enter a positive integer greater than 0 and less than the matrix height (measured in number of buttons) in order to edit the button at this position. The horizontal position can be adjusted with X
No data is altered during this process. It is simple a way of selecting a button for editing in the Button Prefs window.

1.39 QBox: Usage / Button Prefs / Import (Button)

<Y Export (Button)>

Import (Button)

By clicking on this button you will be presented with a file requester asking for an AmigaDos script file. An AmigaDos script file is a file that contains a list of AmigaDos commands (eg. S:User-Startup). Upon selecting an appropriate file, QBox will overwrite the current Command Script (as present in the Button Prefs window) with a converted AmigaDos script.

In short, this button functions as an AmigaDos to QBox button script converter. Ideal for replacing loads of small scripts with one QBox one!

When importing you should be aware that QBox treats the left most semicolon (;) that is not embedded in quotes as the beginning of a comment. As such all text after this semicolon (on the same line) will be ignored. Eg.

Echo "Eg; Horace went skiing" ; A long time ago

Becomes..

Echo "Eg; Horace went skiing"

NOTE: QBox button scripts are batch jobs. They are not true AmigaDOS scripts and therefore have no support direct for script-only AmigaDOS commands (like .key, failat, lab, skip etc). This problem and its solution are discussed in Scripts.

1.40 QBox: Usage / Button Prefs / Export (Button)

<Import (Button) Dim X>

Export (Button)

This button is designed to export and button script to an AmigaDOS script. Simply select a destination file using the standard ASL file requester and the current button script (in the command button gadget) will be sent to the chosen file as standard AmigaDOS.

The file is headed with a comment describing where the script was sourced. Details include button script filename, button name and any script commands used.

1.41 QBox: Usage / Button Prefs / Dim X

<Export (Button) Dim Y>

Dim X

This numeric entry gadget is used to define the number of Buttons wide the Main window matrix is.

The value you enter will always take an absolute value (it will always be interpreted as a positive value) and may only be an integer.

Buttons are normally the width of longest Button Name text. However, if the number of buttons wide combined with button width means that the resulting Main window exceeds the current screen width, button widths are reduced by a multiple of 8 pixels. This ensures that all buttons are displayed, but at the cost of button name text being reduced in length.

This is best avoided by increasing Dim Y if possible, or by using multilayered matrices.

NOTE: The button matrix reloads when the matrix size is altered. This means that the button matrix contains as many of the defined buttons as will fit into this area (without affecting layout). The downside is that you should save the button matrix prior to altering this value.

1.42 QBox: Usage / Button Prefs / Dim Y

<Dim X Menu Prefs>

Dim Y

This numeric entry gadget is used to define the number of Buttons high the Main window matrix is.

The value you enter will always take an absolute value (it will always be interpreted as a positive value) and may only be an integer.

Buttons are normally the 14 pixels high. However, if the number of buttons high combined with button height means that the resulting Main window exceeds the current screen height, the button height is reduced by a minimum number of pixels. This ensures that all buttons are displayed, but at the cost of obscuring all or part of the button text.

This is best avoided by increasing Dim X if possible, or by using multilayered matrices.

NOTE: The button matrix reloads when the matrix size is altered. This means that the button matrix contains as many of the defined buttons as will fit into this area (without affecting layout). The downside is that you should save the button matrix prior to altering this value.

1.43 QBox: Usage / UDM Menu Prefs

<DimY App Window (UDM Prefs)>
Menu Prefs
Hotkey: Right Amiga + M
QBox Button Prefs window
Menu Menu Name HKey HKey
Command Command Script
Copy Exch Clr Undo Rvrt Item Item
New Load Save Iprt Eprt Dim Dim
This window is used for defining user-definable menus (UDMs).
This window is an App Window. That is, if an workbench icon is
dropped on to this window various attributes will be filled in. Click on
App Window for more.

1.44 QBox: Usage / Menu Prefs / App Window (UDM Prefs)

<Menu Prefs Menu Name>
App Window (UDM Prefs)
The Menu Prefs window is an App Window. That is, when a workbench icon is
"dropped" on it, various information from the icon is placed in the window
thereby saving wear on your fingers.
Here, QBox enters the name of the file (to which the icon belongs) and
places it in the Menu Name edit area of the Menu Prefs
window. At the same time it takes the full pathname and filename of the
file and places it into the Menu Command edit area of
the same window.
QBox also recognises multiple icon selections (see Amiga User Guide). The
current menu name will take the name of the first file processed. The
current menu script will take all of the full filenames separating each
with a semicolon (;). The order of the menu script depends solely on the
order in which workbench passes each icons to QBox.

1.45 QBox: Usage / Menu Prefs / Menu Name

<App Window (UDM Prefs) Command Script (UDM Prefs)>
Menu Name
This text entry gadget enables you to alter the name of a chosen UDM menu
item. That is, the visible text part of a menu item under the User menu title.

Choosing a Menu item

To choose a menu item open the Menu Prefs window and then select the required user menu item from the pull-down menus. You should see the details for the selected menu item appear in the Menu Prefs window.

You can use the menu items hotkey, if one exists, for selection purposes.

Altering a Menu

Now that the menu has been chosen, click in the text entry gadget labelled "Menu" (in the Menu Prefs window). Alter the text using the keyboard (cursor keys move the cursor within the gadget). When you have finished making alterations in the text entry gadget press the Return key (the largest grey key). After a major window update you should see that the menu item has changed in the user menu list.

1.46 QBox: Usage / Menu Prefs / Command Script (UDM Prefs)

<Menu Name HotKey>

Command Script (UDM Prefs)

This text entry gadget (labelled "Command") allows an AmigaDOS / QBox script to be associated with a given user menu item. That is, the invisible functional part of a menu item.

Choosing and altering a menu is done in a similar way to altering the Menu Name. However, any changes are not immediately apparent (the menu name will not change). You can, however, verify any alterations by re-choosing the button after editing if you need to.

See Scripts for syntax.

1.47 QBox: Usage / Menu Prefs / HotKey

<Command Script (UDM Prefs) Copy Menu>

HotKey

This text entry gadget (labelled "HKey") allows a single character code to be combined with the Right Amiga key to form a hotkey for the current user menu item being edited.

Simply type in the desired character and it will be assigned. However, if the key combination is already present in the menus you will be told to choose another character.

To check the assignment has gone as planned simply view the user menu list (holding right mouse button over and moving the mouse over "User" in the screen title bar).

1.48 QBox: Usage / Menu Prefs / Copy Menu

<HotKey Exchange Menu>

Copy Menu

Clicking on this button will cause the screen title to change informing you to select a user menu item or press both mouse buttons to abort the copy function.

Selecting a menu item from the "user" menu list will result in the current menu (whose values are in the Menu Prefs window) being copied to the one clicked on.

This is useful for "modelling" menus. These "models" can then be adjusted to suit individual requirements.

Note that it is usual for the hotkey to be abandoned in a copy to prevent multiple menus with the same hotkey.

1.49 QBox: Usage / Menu Prefs / Exchange Menu

<Copy Menu Clear Menu>

Exchange Menu

This button operates not unlike Copy Menu.

Clicking on this button will cause the screen title to change informing you to select a user menu item or press both mousebuttons to abort the exchange function.

Selecting a menu item from the "user" menu list will result in the current menu (whose values are in the Menu Prefs window) being exchanged with the one clicked on. In effect, swapping the two menu items attributes - ideal for rearranging QBox button layout.

Hotkeys are retained.

1.50 QBox: Usage / Menu Prefs / Clear Menu

<Exchange Menu Undo (Menu)>

Clear Menu

This allows you to clear the current menu item definition (the one whose text is visible in the menu prefs window). That is, the menu name and it's associated menu script are set to null (i.e. blanked).

1.51 QBox: Usage / Menu Prefs / Undo (UDM Prefs)

<Clear Menu Revert (UDM Prefs)>

Undo (UDM Prefs)

This button in the Menu Prefs window enables your last menu edit action to be undone. That is, if you edit a menus attributes (name or command) and decide that the new menu items attributes are undesirable, you may, by clicking on this button, force the menus attributes to revert to the values set prior to the edit.

This operation will Undo the following edit events:-

App Window

Menu Name

Menu Command

Clear Menu

Copy Menu

Exchange Menu

Menu Import

Menu Export

Be aware

~~~~~~~~

Undo's will only affect the very last action in the Menu Prefs window. The rules documented in "Be aware" section of Undo (Button) apply equally here. Please read them if you have not done so already.

1.52 QBox: Usage / Menu Prefs / Revert (UDM Prefs)

<Undo (UDM Prefs) Item>

Revert (UDM Prefs)

This button can be seen as either a heavy handed Undo or a Reload option. Both descriptions are applicable here.

Basically, Revert (Rvrt) loads the last loaded/saved menu definition thus undoing any changes and reverting the menu back to its original form.

There is no verification (if there were there would be no benefit over simply using load) so be careful.
~~~~~~~~

1.53 QBox: Usage / Menu Prefs / Item

<Revert (UDM Prefs) Import (UDM Prefs)>

Item

This numeric entry box allows you to see the X-ordinate of the button you are editing (in terms of it's horizontal position in the button matrix).

You may also enter a positive integer greater than 0 and less than the matrix width (measured in number of buttons) in order to edit the button at this position. The vertical position can be adjusted with Y

No data is altered during this process. It is simple a way of selecting a button for editing in the Button Prefs window.

1.54 QBox: Usage / Menu Prefs / Import (UDM Prefs)

<Item Export (UDM Prefs)>

Import (UDM Prefs)

By clicking on this button you will be presented with a file requester asking for an AmigaDos script file. An AmigaDos script file is a file that contains a list of AmigaDos commands (eg. S:User-Startup).

Upon selecting an appropriate file, QBox will overwrite the current Command Script (as present in the Menu Prefs window) with a converted AmigaDos script.

In short, this button functions as an AmigaDos to QBox menu script converter.

When importing you should be aware that QBox treats the left most semicolon (;) that is not embedded in quotes as the beginning of a comment. As such all text after this semicolon (on the same line) will be ignored. Eg.

Echo "Eg; Horace went skiing" ; A long time ago

Becomes..

Echo "Eg; Horace went skiing"

NOTE: QBox menu scripts are batch jobs. They are not true AmigaDOS scripts and therefore have no support direct for script-only AmigaDOS commands (like .key, failat, lab, skip etc). This problem and its solution are discussed in Scripts.

1.55 QBox: Usage / Menu Prefs / Export

<Import (UDM Prefs) Dim>

Export (UDM Prefs)

This button works much the same as Import, but instead of converting AmigaDOS scripts to menu scripts, this converts Menu scripts to AmigaDos ones.

First of all you will be asked to select/define the filename of the new AmigaDOS script using a standard ASL requester. Once done, QBox will automatically convert the current menu script (in the command string gadget) to AmigaDOS writing all output the selected file.

The resultant AmigaDOS script will contain a single comment line on the first line. This comment basically informs any readers where the script was exported from giving details of menu script filename, menu item name and any script commands present.

NOTE: The menulist reloads when the maximum menu size is altered. This means that the menu list contains as many of the defined menu items as will fit into this area (without affecting layout). The downside is that you should save the menu list prior to altering this value.

1.56 QBox: Usage / Menu Prefs / Dim

<Export (UDM Prefs) Misc Prefs>

Dim

This numeric entry gadget is used to define the number of menu items deep the User Defined Menu list is.

The value you enter will always take an absolute value (it will always be interpreted as a positive value) and may only be an integer.

Obviously, the depth of the menu is limitted by the screen height you are working with. To tackle this problem you can use multilayered menu scripts (using ;\M). See Scripts for more.

1.57 QBox: Usage / Misc Prefs

<Dim I/O Button>

Miscellaneous Prefs

Hotkey: Right Amiga + N

QBox Misc Prefs window

I/O Buffer I/O Button Size I/O Menu I/O Menu Size

Win Title Window Title

About

This small interface is neatly tucked away under the Prefs/Misc menu item.
It is separate from the Button Prefs window as it has nothing to do
with buttons.

1.58 QBox: Usage / Misc Prefs / I/O Button

<Misc Prefs I/O Menu>

I/O Button

This numeric edit gadget enables you, the user, to set the I/O buffer size
used to read button preference files. That is, you can set how much data
will be accepted on every file read. The larger the buffer, the more room
for lengthy button scripts.

The value defaults to 8192 which gives about 8kb for each button script
line (less other button details). The minimum allowable value is 160 and
the maximum is 65535. However, as the I/O Buffer increasing in size so
does QBox's instability. This is a problem rooted in Blitz2 that I am
currently having trouble solving. Suggestions are welcome.

A safe upper limit is about 16384. Experiment with caution!

ALWAYS: Ensure that the length of any given line in the prefs file does
not exceed the I/O buffer size. You may end up with some odd behaviour
otherwise. You can use GetSize (part of the TLB-Tools distribution)
to check this if you are unsure of this fact.

1.59 QBox: Usage / Misc Prefs / I/O Menu

<I/O Button Window Title>

I/O Menu

This numeric edit gadget enables you, the user, to set the I/O buffer size
used to read menu preference files. That is, the maximum number of
characters read/written during file I/O.

The value defaults to 8192 giving upto 8kb for each line in the menu
script. The minimum value is 160 and the maximum 65535. Take care as you
increase this problem, as QBox becomes more unstable the larger the
value used.

A safe upper limit is about 16384. Experiment with caution!

ALWAYS: Ensure that the length of any given line in the prefs file does
not exceed the I/O buffer size.

The same rules and cautions apply to the I/O Menu buffer as those stated
in I/O Button.

1.60 QBox: Usage / Misc Prefs / Window Title

<I/O Menu ToolTypes>

Window Title

This text entry gadget allows you to define the Main windows title bar text.

Simply enter your text and hit return. The main window will update and you title text will appear. That's all there is to it really.

1.61 QBox: Usage / ToolTypes

<Window Title Contact>

ToolTypes

QBox supports icon tool types (see Amiga User Guide) that may be edited from the icon information screen. Currently, the following tooltypes are supported:-

DONOTWAIT

This tooltype prevents other programs being held up until QBox has finished during system boot up. If QBox is present in your WBStartup drawer you should have this present in QBox.info. Technically speaking DONOTWAIT is handled externally to QBox by workbench 2.x/3.x.

CX_POPUP=Yes|No

This tooltype takes a single argument that maybe either YES or NO (eg. CX_POPUP=YES). This icon tooltype simply defines whether or not the QBox interface is visible when the program is started or not. If "CX_POPUP=NO" QBox will run in the background until "Enable"/"Show Interface" is clicked on in the commodities Exchange program, or until you press the QBox hotkey (default: ctrl lcommand z).

(See Amiga User Guide)

CX_POPKEY=hotkey

This icon tooltype is used to define what keys force QBox out of hiding (see Hide and CX_POPUP above). Several examples follow:

CX_POPKEY=ctrl alt z

CX_POPKEY=rcommand ralt del

CX_POPKEY=ctrl lcommand z

BUTTONFILE=filename

This is used to define the path and filename of a QBox button prefs file. The CLI BUTTONFILE argument overrides this tooltype if defined.

Eg. BUTTONFILE=Startup.QBox

MENUFILE=filename

This is much the same as BUTTONFILE (above) only this is used to define the path and filename of a QBox menu prefs file.

CLI MENUFILE argument overrides this tooltype if defined.

Eg. MENUFILE=StartMenu.QBox

TIMEOUT=secs

The tooltype equivalent of TIMEOUT=secs CLI argument. This tooltype is used to define the amount of time (in seconds) that QBox can be left inactive before quitting. The CLI TIMEOUT argument overrides this tooltype if it is defined.

Eg. TIMEOUT=20

DEFFILE=filename

Another CLI equivalent, this time of CLI DEFFILE=file. If the timeout feature is used (either through the tooltype or CLI TIMEOUT=secs) and QBox is left to timeout, then the AmigaDOS script file defined here will be executed. CLI DEFFILE=file overrides this tooltype if it is defined.

Note: The AmigaDOS script can be headed with ;\RNXH tags on the first line of the file so that execution can be controlled.

Eg. DEFFILE=S:Startup-Sequence

where S:Startup-Sequence's first line reads ";\RN" (no quotes).

SHOWGUIDE=filename

This is used to define multiview/amigaguide/ppguide path and filename that is used when Help is selected from the QBox menus.

Eg. SHOWGUIDE=Dh0:utilities/MultiView

GUIDE=filename

This is used to define where the QBox AmigaGuide (tm) resides that is used when Help is selected from the QBox menus.

Eg. GUIDE=Locale:help/english/QBox.guide

MAINX=x ord

MAINY=y ord

These two tooltypes are used to define the position of the Main window. x ord and y ord should be replaced with X and Y screen co-ordinates respectively.

Eg. MAINX=0

MAINY=20

BUTTONX=x ord

BUTTONY=y ord

These two tooltypes are similar to MAINX and MAINY, only this pair define
the window position of the Button Prefs window.
Eg. BUTTONX=360
BUTTONY=196
MISCX=x ord
MISCY=y ord
This pair of tooltypes define the position of the Misc Prefs window.
As with MAINX, MAINY, BUTTONX and BUTTONY this pair should be supplied with
the window position co-ordinates.
Eg. MISCX=0
MISCY=196
MENUX=x ord
MENUY=y ord
This pair of tooltypes define the Menu Prefs window position. As
with the other "position" tooltypes you should supply this pair with the
required window co-ordinates.
Eg. MENUX=398
MENUY=14
Note: MAINX, MAINY, BUTTONX, BUTTONY, MISCX, MISCY, MENUX and MENUY are
used to define the top left corner of Main, Button and Misc preference
windows. The top left of the screen is x=0, y=0. Value limits are x=640
and y=512. Dont't worry so much about limits QBox will move windows until
they are in view.

1.62 QBox: File formats

<ToolTypes Contact>
File Formats
Button Scripts
The first entry (or header) of a button script is used to define matrix
details and other properties fixed for the entire matrix:-
DimX,DimY,Window_Title,nnnnn A C P
[Links above show editor text that can be applied at least in part here]
where:-
dimx is the width of the button matrix measured in buttons
dimy is the height of the button matrix measured in buttons
window_title is the text that appears in the window drag bar
nnnnn is the i/o buffer size measured in characters
A turns off autoabout (but not forced auto-abouts in EVALUATION version)

C turns off case sensitivity of buttons

P turns off prefs button (REGISTERED vX.XX version only)

example:-

1,2,Example,1024AC

means 1x2 matrix

Window drag bar contains "Example"

Maximum line length of 1024 allowed

Auto-abouts off

Case sensitivity off

Prefs button on (because of it's omission in the header)

All lines after this are used to dictate individual button properties:-

X,Y,Button_Name,Button_Script

[Links above show editor text that can be applied at least in part here]

where:-

X is the button X ordinate (1=left most column, DimX=right most column)

Y is the button Y ordinate (1=top row, DimY=bottom row)

Button_Name is the text displayed in the button gadget

Button_Script is the functional part of the button (see Scripts for more)

example:-

So with the example for the header we could have

1,2,Example,1024AC

1,1,Top Button,Echo "Top Button pressed";\RH

1,2,Bottom Button,Echo "Bottom Button pressed";\RH

which would appear thus (only using gadtools!):-

_____

|º|Example____|=|

|__Top Button___| -> Top Button Pressed

|_Bottom Button_| -> Bottom Button Pressed

Note: The button_script may contain commas.

Menu Scripts

The first line of a menu script file is concerned with the number of user menu items present in the list. A second field is concerned with the I/O buffer size for menu scripts.

Menu_Items,Menu_Buffer

[Links above show editor text that can be applied at least in part here]

where:-

Menu_Items is the number of menu items (lines) in the file

Menu_Buffer is the maximum length possible for menu script lines

example:-

3,8196

means Add 3 menu items to the user menu

Max menu script line length of 8196

All lines after the header define actual menu items and their properties:-

Menu_Text,Hotkey,Menu_Script

[Links above show editor text that can be applied at least in part here]

where:-

Menu_Text is the text displayed in the "User" pull-down menu

Hotkey is the key that, when pressed with the Right Amiga key, is used as

a keyboard short cut for the menu item.

Menu_Script is the functional part of the menu item (see Scripts for more)

example:-

So with the example we had for the header line we could have

3,8196

Hello,1,Echo "Hello";\RH

Other Menu,2,S:AlternateMenu.QBox;\M

MultiView,3,Run >NIL: Dh0:utilities/multiview Screen;\RN

which would appear in the pulldown menus thus:-

_Project_____User_____

| Hello A1 |

| Other Menu A2 |

|_MultiView___A3_|

1.63 QBox: Contact

<File Formats Credits>

Contacting the Author

The author may currently be contacted for commercial fee negotiation at

the following address:-

T.L.Bullock

c/o 74A Shilton Road,

Carterton,

Oxfordshire OX18 1EL,

England.

Any enquiries pertaining bug reports, new versions, problems or other

programs by the same author may be obtained at the above address or by

emailing (valid until Dec 1995 - email address may change there after):

e0190162@brookes.ac.uk

1.64 QBox: Credits

<Contact History>

Credits

QBox was designed, implemented and documented by the author and copyright
holder T.L.Bullock.

About the author

Name: Tony Bullock

Age : 23

Stat: Married

Aka : hAVoC / Mariner

Occup: Between uni and work

A special thanks must go to the following people:-

Acid Software Creators of Blitz Basic 2 - this would not
have been possible without them. Shame about
the support tho.

Steven Matty & (of Leading Edge Software) for creating
Stephen McNamara RIBlitzLibs (©1994,95) and supporting it's
users (me 4 1 =8O).

Thanks also for the help concerning various
blitz/RI functions - its very much
appreciated.

Blitz coders: get RIBlitzLibs now!

Richard T. Elmore (of HeadSoft Software) for creating ElmoreLibs
(©1994). Nice.

Blitz coders: get this too!

Liz For putting up with the hours of programming.

1.65 QBox: History

<Credits Future>

History

Newest first:-

v1.14 Update: ToolTypes work properly (thanks Leading Edge)

CLI commands now match tooltype names (saves confusion):-

BUTTON -> BUTTONFILE

MENU -> MENUFILE

SECONDS -> TIMEOUT

DEFAULT -> DEFFILE

Bugfix: TIMEOUT function now works.
Note: Only a minor update due to lack of free time
ToolTypes fixed with new tooltypes.obj from Leading Edge

---

v1.13 New: Rvrt (Revert) in button&menu prefs for reloading scripts
Eprt (Export) in button&menu prefs allows script to amigados
/H QBox command for pausing output
Update: Embedded semicolons in quotes are not command seperators
Port changed to Iprt (Import) (key short-cut now "I")
Verification when UDMs are NEWed
Load code altered slightly (internal)
Default matrix settings improved
Bugfix: Window titles now update properly
Load and Save "talk" to prefs eds better (internal)
Menu copy is UNDOne hotkey
Menu exch now UNDOes properly (was really bad)
Cancelled SaveAs no longer spoils DimX/Y updates
Menu hotkeys now work (simple fix)
AppWindows now work (DEFLIBS trouble made problems in v1.12)
Note: v1.13/μ/e and above use conditional compiling. This means
I should be able to spend more time authoring new QBox
features rather than messing about with updating each
individual sub-version code (hopefully). Say "Hurrah!"
v1.12 New: Case Buttons menu item for key short-cut case sensitivity
Delete menu item for deleting unwanted scripts/files
\B(\L) and \M can now be used together as \MB or \BM
Shareware registration scheme incorporated (encrypted data)
I/O Menu buffer for control menu script file I/O
Update: \L now \B as in \B(utton) and \M(enu) scripts
CLI FILE argument changed to BUTTON
Commas can appear in functional part of button/menu scripts
Bugfix: Menu/button prefs editors are safer (more validation)
File loading code improved
Notes: Tooltypes still on ice, but new features (below) are
coded ready for when i can safely include them.
Decided QBox will be released as shareware.
New*:BUTTONFILE tooltype for defining default button prefs file
MENUFILE tooltype for defining default menu prefs file
TIMEOUT tooltype for defining idle time before auto-quit

DEFFILE tooltype defines amigados script ran on auto-quit
Tooltypes will run before cli parser so defaults are set
v1.11 New: Simple User Menu editor (inc. appwindow and import)
Load and Save menu preferences
MENUX/Y tooltypes ready to roll when tooltypes debugged
Simplistic recognition between button and menu prefs files
MENU CLI argument fot specifying menu prefs
All windows now use screen title bar
Update: Copy & Exch pref buttons prompt with title bar not req
Project menu now covers new menu load,save etc functions
Prefs button now opens and closes all prefs windows
Bugfix: Various small bug fixes
Note: Still no ToolTypes support
Rewrite of μ vers to cover menu loading, new cli etc.
Micro version is more important now as full version has
out grown its role as a small launch program. Consider
the full version as a prefs editor.
v1.10 New: Key shortcuts for all buttons in all windows
Screen title reinstated and popkey combi displayed
Update: Popkey now works properly (Many thanks Leading Edge)
Windows are activated more intuitively
BugFix: V.Minor fix to hide code (shared var was missing)
Note: ToolTypes are still on @!**&% ice!
Micro (μ) version re-written including auto-displaying About
v1.09d New: Added timeout system for use in s:user-startup
SECONDS/K/N CLI argument added to control timeout period
DEFAULT/K/N CLI argument added to define timeout scriptnames
Prepend QBox commands to DEFAULT file
Update: Improved CLI parser
Main process loop slightly more efficient
Title updates without closing and reopening the main window
v1.09c New: Clear (Clr) button in the button prefs for blanking a button
/S QBox command added for scripts contains script-only cmds
Hide option in Quit requester
Button names may now have leading/trailing blanks
Bugfix: User menu limits properly
Save should now save into the correct directory
X/Y may never DimX/DimY (there was an exception before)
CLI argument sets ASL requester fields properly

v1.09b New: MultiView default now include FONTNAME and FONTSIZE options
Bugfix: Help option should work properly
ReAdded: Commodity support (no hotkey)
AppWindow support
Reason: From what I can tell, v1.09 crashes when ever ToolTypes
code is used. Commodity and appwindow support seem to be
okay.
Note: RILibs is back in business. ToolTypes and Hotkey support
will be added into QBox as soon as RILib allows for it.
v1.09a Removed: Commodity support
AppWindow support
ToolTypes support
Reason: It appears that Blitz fails to free memory properly (in
compiled versions) after QBox quits when the above features
are used. I could live with this normally, but this problem
with freeing memory results in a software error/guru
(81000005 0001AE28).
As soon these problems are fixed (and i'm sure they will)
QBox will have the removed features added.
Note: RILibs is on hold until the above troubles are fixed.
v1.09 New: Copy buttons
Exchange buttons
Import AmigaDos script to QBox button script
Undo last button config operation
Goto X/Y matrix position input area
Update: X label now DimX in button prefs
Y label now DimY in button prefs
Note: From this version on, all QBox distributions will contain
two versions: a standard vx.xx and a micro vx.xxμ. The
micro version is a reduced version without Prefs editors or
any associated operations (i.e. save, new, appwindow etc)
v1.08 New: QBox is now a commodity (no hotkey yet)
Icon tooltypes supported
User notification and verification requesters
Hide menu option added
Note: ElmoreLibs will be used from here on with RILibs
v1.07 New: Button prefs window is now an AppWindow
Utilises CatchDosErr requesters
Note: RILibs will be used from here on until other wise stated

v1.06 New: Rudimentary user-definable menu system

On-line help menu item

Small font used when button height gets tight

Update: Improved hotkey layout

v1.05 New: Window (sizes) now sensitive System font in title bar

Menu disabled when About window visible

Both prefs windows, when active, stay on page/screen

Update: File format changed slightly for more intuitive edits

About text shifted about to make it easier to read

Fonts changed

Better internal font handling to make future update easier

New maximum of 60 chars for button text

New minimum of 160 for I/O buffer

BugFix: Main window stays on screen properly when Y updated

Window coords update properly after window position change

Auto-button width debugged

Main window stays on screen properly after button width upd

Misc: MagicWB-like icon added

v1.04 New: Windows now open where they were last closed

Main window stays on screen when updated winsize/scrnmode

CLI help text available (type ? as first CLI argument)

BugFix: Button prefs opens safely now

Filenames update in ASL filerequester properly

v1.03 New: Reduced button sizing overhead (slight speed increase)

Adjusted prefs file order so manual edits more intuitive

Removed: Main window resizing (maybe re-included in future)

BugFix: post-load button prefs windows linked to button properly

v1.02 New: Automatic button width update

During loading Main window only refreshes if successful Load

Removed: Update menu option (now automated)

BugFix: Both prefs windows update after loading

v1.01 New: Misc Prefs window

Easy access to main window title

Easy access to I/O buffer size

Pull-down menus

Update button size option

Hotkeys

Fitting all buttons on screen

/F QBox command for loading matrix

Bugfix: Error handler no longer gurus!
v1.00 First Release
Pre QBox is based on a menu system I wrote way back in 1990 as part
of an (over?) ambitious AMOS project. The original used a 640x256
(16 colour) screen with hard coded graphics looking not unlike
MenuMaster (remember that?). Unfortunately, the original code has
been lost in the depths of the space-time continuum (the disks it
was on have probably been formatted, corrupted or both).
Needless to say QBox is a total rewrite!

1.66 QBox: Future

<History Restrictions>
Future
* Iconify / Menuify QBox (if I feel the need)
* Hotkeys for all edit boxes (ActiveString doen't work with GadTools)
* Window for runtime command parameter selection (with editor & on/off tog)
* Proper button updating (GTSetAttr doesn't seem to support this)
* Proper menu updating (Currently each window has to be re-attached)
* Proper font sensitivity for buttons etc
* Force command to the last screen & window used (for CD32 keyboard faking)
* Scrolling Main window so that 1000s of buttons may be held in one script
* Default button prefs stored as icon tooltypes (maybe)
* Localisation (ie. language catalogues)
* Release execution CON: window from QBox [1]
* Detach QBox from CLI [2]
* 8svx Sample-Button association [3]
[1] This can be achieved currently by prepending commands (that don't
automatically terminate before the next command/end of button script)
with "Run >NIL:".
[2] Can be achieved by typing Run >NIL: QBox or by starting QBox from
workbench icon.
[3] Currently, achievable by running a sample player >NIL: at the
beginning of a button script.
Notes
QBox was coded and compiled in Acid Softwares Blitz Basic 2. With any luck
an Amiga E version will be appearing (as soon as I've registered and learnt
the language!). That isn't to say i'm not going to use Blitz any more tho.
=8O) <-- Angle head -90 to floor (IRC users understand) *GRIN*

1.67 QBox: Restrictions

<Future Known Bugs>

Restrictions

QBox has several restrictions. Restrictions proceeded by - can not be altered, they are restrictions imposed by Blitz or Commodore. + means I can adjust the restriction for REGISTERED users if required - simply contact me.

- I/O Buffer may be a value 160 to 65535 (warning: after 16384 things can get a bit unstable).

+ Button names may only be 60 character long. I will happily alter this on request, but the lower I/O buffer limit will have to increase.

- The work area for QBox (ie. all windows) is confined to 640x512 (sob). Hopefully, Acid will have fixed WB size commands in BUM.

- The number of buttons in the main button box should not be allowed to shrink less than a letter (when main window exceeds screen width buttons lose N characters). Why? Because button labels vanish, window refreshes become ridiculously slow and because QBox might just guru. Be sensible, use filename;\B for loading QBox prefs. I may add scrolling at some point to avoid this one.

+ Window titles are restricted to 80 characters long

- QBox steals a bit of memory when closed. This is fixed in BUM6 and only needs a recompile under that version (so i'm told).

+ QBox time-out period may not be less than 1 second (as less than this would make QBox virually impossible to use).

In addition EVALUATION versions also have the following restrictions imposed on them as an incentive to register:-

* No preference editors

* Button matrix may only reach 4 buttons wide by 4 buttons deep

* User Defined Menus may only hold 3 menu items

* Occasionally a forced sticky About window appears that stays on screen 5 seconds after clicking a mouse button.

* Menu and Button I/O buffers may only reach a maximum of 1024 characters (restricting script definition lengths)

1.68 QBox: Known Bugs

<Restrictions Index>

Known Bugs

* QBox attempts to size its main window according to workbench dimensions.
Unfortunately, (as far as I can tell) Acid chose to omit Super Hires
and many other monitors. BUM 1-5 functions seem to only return dimensions
upto 640x512. Not good!
I must say, this bug appear to be hang-overs from Acids Blitz2. This, to
my mind, is not on - particularly when you have to subscribe to BUM for
any fixes that might arise. I would like to say that Acid have found a good
system of upgrading, but I'd be telling a gross lie. Taking a look at
other companies (Europress, Softwood, EA and many others beside, who either
make updates available via PD libraries/ftp/coverdisks or by charging a
nominal update fee) it becomes obvious that Acid aren't really taking
Blitz user support seriously. I don't stand alone in this view; I'd go so
far as to say most Blitz users are disappointed by the lack-lustre support.
Come on Acid, you have one of the best BASIC packages available, let's see
some support to match (and a more comprehensive up-to-date manual too).
Please...

1.69 QBox: Index

<Misc

Index

Acknowledgements

Button prefs, App window*

Button prefs, Button name*

Button prefs, Clear*

Button prefs, Command script*

Button prefs, Copy*

Button prefs, Dim X*

Button prefs, Dim Y*

Button prefs, Exchange*

Button prefs, Export*

Button prefs, Import*

Button prefs, Load*

Button prefs, New*

Button prefs, Revert*

* Only available in the REGISTERED vX.XX version

+ Link only available to REGISTERED users