BlitzGUIGen

COLLABORATORS			
	TITLE : BlitzGUIGen		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 22, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

BlitzGUIGen

1.1 BlitzGUIGen.Manual

```
                  Welcome to BlitzGUIGen
                Version 1.8 : Creation June 1997

           GadToolsBox .gui file conversion utility.

                Legal  Notices and Copyrights
                What Is...
                Requirements
                An  Overview
                Tooltypes
                CLI Parameters
                Installing Key File
                In Depth
                Contacting the Author
                Registration  Details
                History of Releases
                Reporting Bugs
                People to Thank
                Things Still To Do
```

1.2 Legal Notices and Copyrights

```
The following sections contain legal information about
specific programs mentioned in this file. Please refer
to the relevant section:

                        BlitzGUIGen
                        GadToolsBox
                         IntuTools
                        PowerPacker
                         Decrunch
```

1.3 Legal stuff about BlitzGUIGen

1.4 Legal stuff about GadToolsBox

1.5 Legal stuff about IntuTools

1.6 Legal Stuff about PowerPacker and Decrunch

1.7 What is that BlitzGUIGen thing?

BlitzGUIGen is the program you've been looking for all this time!.

If, like me, you're a serious user of Blitz Basic 2 and you do a lot of Intuition programming, then, up till now, you've been doing it the hard way. I have spent many hours programming Graphic User Interfaces manually, compiling the source, moving gadgets one pixel left, re-compiling, moving, re-compiling. Get the message!

Now I do it the easy way, BlitzGUIGen!

If you've ever used GadToolsBox, you'll know how easy it can be to generate professional looking front ends for your programs. To be fair, IntuTools does try to come to some sort of a compromise, but just doesn't make it into the 'big league'.
Having noticed that GadToolsBox generates source for you, I needed to do something that will generate source for me in Blitz Basic. That's where BlitzGUIGen comes into its own. Everything that is supported in GadToolsBox and Blitz Basic has been included, even nm_BarLabel menu separators (using the RIGTMenusLib.obj), and backfill patterns.
See the Overview section for details on using BlitzGUIGen.

1.8 What requirements do I need?

I suppose the main requirement for BlitzGUIGen to be of any use to you, is Blitz Basic 2. You will require at least version 1.9 as this is the version I am currently using and some updated commands included in version 1.9 are used. Versions less than 1.9 may still be used with BlitzGUIGen but some source may be incomplete.

Secondly, you will need Version 2.0 of GadToolsBox. Lesser versions of GadToolsBox will not work with BlitzGUIGen, as files are checked for GadToolsBox version number before being converted. Later versions may be supported in the future once I upgrade my GadToolsBox version.

You will also need the Decrunch executable that accompanies the PowerPacker distribution, if you want to access any crunched .gui files made with GadToolsBox.

Of course, it should go without saying, that GadTools gadgets are only available in Kickstarts 2.x and higher. You will need to have at least KS2.x up in order to use this software.

You will also need a copy of Reqtools.library in you're Libs: assign.

Once these requirements are met, you are then equipped to use BlitzGUIGen.

One thing that you must make sure is that the directory 'Blitz2' is assigned to 'Blitz2:', otherwise BlitzGUIGen will not be able to

launch Blitz Basic's editor to load in source.

That's about it for 'what do I need'. See the Overview section for details on using BlitzGUIGen.

1.9 An overview on using BlitzGUIGen

BlitzGUIGen is extremely easy to use, and has been designed to be as simple as possible to operate. A few subjects may need to be covered and are explained below:-

Installation

If you wanted to do this the easy way, then you could use the installation software included in this archive. Alternatively, you can install it by hand if you don't trust me!.
Start by selecting somewhere to put the software, or create a new drawer for it.
Next, copy the executable file BlitzGUIGen, and its icon BlitzGUIGen.info into your selected destination.

I have a sub-drawer where I keep all my GUI files called 'GUI-Gen', something similar would be a good way of keeping things tidy.

Should you need it, the 'Decrunch' executable should be placed into your C: assign.

Set Up

Now you have the option to set up the tooltypes for BlitzGUIGen. I would suggest reading the Using ToolTypes section for more information and descriptions on achieving this, or the CLI Parameter Passing section if you intend to run it via a shell.

Installing Key File

Once you have registered your copy, you will receive a Key File, which should be installed in the same directory as BlitzGUIGen itself. The Key File will now unlock your version, and will enable the following functions:

- Personalize your copy by showing your name and number in title
- Personalize your copy by showing your details in 'About' requester
- Enable the keyboard shortcuts
- Enable the 'Load New Blitz' button
- Enable the use of crunched gui's
- Remove the automatic 'About' requester on open and close
- Put your name and number in source
- Enable the generation of custom screens
- Enable the creation of menu lists
- Enable the creation of a .xtra file
- Enable the use of MAKETOGGLE tooltype
- Enable the use of ACTIVATESTRING tooltype

```
    Enable the use of FUNCTIONS tooltype
    Enable the use of BACKFILL tooltype
```

Executing

Firstly, after loading up the program by double-clicking its icon, or entering BlitzGUIGen (params) from the command line of the CLI, you will be presented with a simple GUI.
For information on CLI parameters, refer to the section Parameter passing from the CLI for details of these, or Using ToolTypes for info on these.
(By the way, this GUI was created with BlitzGUIGen and GadToolsBox!).
If you need to alter any program parameters, you should do so now, before converting any .gui files.

The set of five checkboxes at the top left side of the window are to allow you over-ride the default settings.

Starting from the top, is the 'Screen Code' gadget.
This will allow you to decide, at this point, whether or not to create screen data within your source.
If this is not selected, then the source will not compile straight away from the Blitz 2 editor, but is intended, instead, to be used as an include file that creates your designed GUI.

To the right is 'Backfill'. This checkbox will enable/disable the ability to give your gui a backfill pattern. This is much like the grey background pattern you see in system requesters.

Next, working down, is the 'Menu Code' gadget. This, being similar to the above, will, if you have created menus, allow you to create them within your Blitz source.

Next is the 'FillColour' gadget. This specifies the primary colour that any backfill pattern will appear in. It defaults to 2, but can be changed more permanently by using tooltypes.

The 'Make Source Use WB Screen' gadget comes next, and this will force your source to use the Workbench screen as the default screen that your window will appear on. De-selecting this will use the front most screen instead.
As of version 1.3 +, this selection will over-ride any screens that have been created from within GadToolsBox, and force your window onto the Workbench screen.

Lastly, is the 'Create Event Handler' gadget. This will create an IDCMP event handler for you to handle things like gadget hits and menu hits etc. The handy thing here, is that if you have made some meaningful labels for each of your gadgets, then these labels will be used in this loop, which makes it a lot easier to work out which ID number goes with which gadget.

The default DetailPen and BlockPen can be altered from the string gadgets over on the top right side.
Once these are as you want them, you are ready to continue.

Conversion is a simple button press. The 'Select Source GUI...'

button allows you to select a GadToolsBox file from a file requester for conversion from the selected path. The file requester will only show .gui files, which is the format for conversion.

Once the selected file has been parsed by BlitzGUIGen for correct version, the GUI will show what is happening. The software will tell you once the source has been created, and you can elect to either quit, using the quit gadget or the close gadget, or load a new Blitz editor screen with the source ready to view and/or compile.
If the file already exists upon trying to save the source, you will be presented with a Save requester which will give you the opportunity to rename the saved source's name. Should you try and save a file with the same name as an existing one, you will be warned by the file requester reappearing and giving you the opportunity to change the name.

Understanding created source

The created source should be straight forward enough to understand without too many complications.
BlitzGUIGen will define some variables in your source that may appear confusing to start with.

First, let us assume you created a project in GadToolsBox with the name 'MyGui'. This is the label you give to the project from the 'Edit Data' item from the 'Window' menu of GadToolsBox. This label will be used in creating your Blitz 2 source.

Working from the top of the source, you will see the copyright message and version number created by BlitzGUIGen.

Next, if you have elected to create screen code, will come the sub section of code that will define the screen.
Then a label 'InitialiseVars' will start a block of variable declarations.
MyGuiWnd = ???, will be the first variable you see. This variable will normally have a value of zero, but may be changed later.
See the 'WINDOWLABEL=' tooltype for info on changing this value.
This will be the number of your window in your source, which gadgets etc. will be attached to.
Next will be 'MyGuiGList = 0' which is the gadgetlist number in your source. You can manually change this value in your source, or refer to the 'GLISTLABEL=' tooltype to change permantly.
Lastly, 'MyGuiScr = 0' will be the number of your current screen. This, also could be changed manually, if desired, or permantly with the 'SCREENLABEL=' tooltype.

If you have created any menus, 'MyGuiMList = 0' will be next. This defines the number for the menulist. Again this can be changed manually later on, or permantly with the 'MENULABEL=' tooltype. Then a 'Gosub xxxMenu' or 'xxxMenu{}' will follow which will jump to a subroutine in you're source which will define your menus.

Next comes your list of gadgets, which are all standard Blitz commands, and are described in detail in the Reference Manual that comes with Blitz Basic 2. Some gadgets may have attached tags, using the GTTags

command. This will affect your gadget in a way not directly supported in Blitz 2.

Your window is defined next, as again, a standard Blitz command.

Should you have elected to create a backfill pattern, the code for this will be either above or below here. This should be reasonably straight forward and easy to understand. The window's structure is grabbed first as this is used in building and creating the backfill. The 'pens' are specified next and the primary pens is the one that is changable from the main gui.
You could change the value of the secondary pen as well. Try out different combinations of values for these, you can get some nice MUI style backfills.
Then the backfill pattern is blitted into the window.

You may have created bevel boxes in your window. If so, there will be one line of code for each bevelbox next. Each line will draw a box at the same position as the bevelbox in order to clear it to background colour. This way anything inside the bevelbox is not disturbed by the backfill pattern, and helps to give the window a more system like look.

The bevelbox's, if any, will be drawn into the window next.

If gadgets have been defined then the 'AttachGTList xxx,yyy' command is next, to attach them to your window.

Again, if you have created menus, then these will be attached next using the standard SetMenu command.
A RastPort command may be next, depending on the presence of 'IntuiTexts', then, if present, will be rendered.

The main event handler will be next, and this aims to make it as easy as possible to test your GUI. You may notice that the only way to exit from your GUI once compiled, is to hit the close gadget, if you have one. If not, a panic feature is included and pressing 'ESC' will break you out immediately. At least this way, you will be able to test the action of all the gadgets without exiting back to Blitz 2 straight away.
As mentioned elsewhere, meaningful gadget labels should be used from within GadToolsBox as these are used from within the event handler loop. This makes it easier to check for specific response to a gadget rather than referring to only the ID number.
A similar loop will follow that for the menu event handler, and then another similar one for keyboard shortcuts.

Another subroutine will follow, which forms how you quit. Currently, it just quits, but you can expand on this to create custom exits, like quit requesters etc.

If menus are present then these will be defined in the following subroutine, which, are again, all standard Blitz commands. One note here, is that BarLabel menu separators are not created here, even if they are present in the GUI file, because the standard Blitz menu commands do not, as far as I know, support this function. As of Version 1.6+, GadTools Menu's can be created via the library 'RIGTMenuLib.obj, and as a result, now supports nm_BarLabel separators.

One final note is that you may notice a small graphics routine right at the end of your source. This will be to create a 'GetFile' gadgets images, if they have been used. Although this is freely usable by you in your source, I wouldn't recommend altering any of the values in it, it could lead to unpredictable results, and you are advised to leave well alone. I will apologize for this rather 'messy' way of creating 'GetFile' gadgets, but it seemed an easy way for the time being. Look at it as Blitz's way of implementing BOOPSI! ;0)

As of version 1.7, the ability to create Statements rather than subroutines has been implemented, and if this has been selected, these routines will be first in the listing rather than after the main loop. The FUNCTIONS tooltype outlines this further.

1.10 An In Depth look at BlitzGUIGen

This version of BlitzGUIGen will generate the following types of GUI components to V37 (2.04) standards:

GetFile — These are the gadgets that you may see in a GUI that includes choosing a disk path or volume. They are actually GTShape gadgets with a little bit of trickery involved.

GTButton — These are the buttons you see all over the place. A typical example is the buttons you see on system requesters. GadTools buttons allow you to underscore a certain character on the face of the gadget to act as . a keyboard shortcut

GTCheckBox — These enable the user to switch some item on or off. a small box appears that when clicked on, a small tick will show that it has been switched on.

GTInteger — These are similar to GTString gadgets but accept only whole number inputs instead of alpha characters.

GTListView — These are the gadgets you see that allow you to read or enter an amount of different items. A typical example here would be the selection of printer you want to use in the printer.prefs window.

GTMX — These are a set of radio buttons of which only one can be on at any one time (Mutually eXclusive). They allow the user to make a selection from a choice of a few. This is just a different way to display a GTCycle gadget.

GTNumber — This is a read only number gadget, which is used to inform the user of some numeric value.

GTCycle — These are the button gadgets you see that change their faces as you click on them. This allows for a selection to be made from the gadget.

GTPalette This is the colour palette editing gadget that you see every where in preferences sections. Its use should be fairly obvious.

GTScroller These allow the user to scroll through a list of items etc. Similar to the slider attached to a GTListView gadget.

GTSlider This allows your user to set a level of some sort. Normally used for volume or intensity of colour etc.

GTString These are gadgets that allow you to enter a string of characters into the machine. A typical application for these is in a database program to enter in details. The beauty of these is that an optional label can be assigned to the gadget, which can also be underscored.

GTText These are similar to the GTString gadgets but are read only. You cannot enter text from here. A status display would be an ideal example for using these.

Screen This will be the screen that your window will open on. This could be Public, Custom or even the Workbench screen.

Window This one doesn't really need explaining, I hope. All your gadgets will be rendered into this window once it has been opened.

Menus These also shouldn't require explaining. The menus that have been created will be attached the window once opened. nm_BarLabel menu separators are not supported by these 'standard' menus.

GTMenus As of Version 1.6, you now have the option to create GadTools Menu's. This requires that your 'DefLibs' file contains the commands within the RIGTMenuLib.obj. This is included in the distribution, and you should refer to the 'Install_RIGTMenuLib.readme file. Thanks to Steve McNamara of Leading Edge Software for his great library.

BevelBox These box's are actually borders, similar to the borders around button gadgets. They can be used to split up the window into certain parts, so that the gadget layout makes more sense to the user. The option to recess or dropbox these box's is very handy indeed. Should you have elected to create a backfill for the window containing bevelboxes, the areas inside the boxes will be cleared giving the impression of depth.

IntuiText This is text that is rendered into the window once it has been opened. Different colours are allowed and are supported by BlitzGUIGen.

Backfill A backfill pattern is a coloured background that appears in a window apparently behind everything else. It

```
            makes the window stand out and be noticed more easily, and
            is good to use for important information or requests.

Please Note: Custom screens created by GadToolsBox are currently
            not supported by BlitzGUIGen prior to version 1.3, and
            the structure is ignored.
```

1.11 Using tooltypes from Workbench

```
So far the supported ToolTypes are:

NOMAIN
This is a switch that allows you to turn off the generation of
the main loop that handles IDCMP events. Please note that with
this switch set, the code that is generated will not compile
directly from the Blitz Basic 2 editor, and is now meant to be
included as a sub-routine.
This is the same as the CLI parameter '-e'.

NOSCREEN
This switch allows you to turn off the generation of code that
assigns a screen for your window to open on. Please note that
with this switch set, the code that is generated will not
compile directly from the Blitz Basic 2 editor, and is now
meant to be included as a sub-routine.
This is the same as the CLI parameter '-s'.

PATH= path
This tooltype will allow you to specify a path for BlitzGUIGen to
look at when searching for .gui files. path may be any
legal AmigaDOS path:

eg. sys:GUI/blitzgui

When the filerequester opens, it will now be looking into this
directory for .gui files. Path will also affect the
path to save source files to. If SAVEPATH= hasn't been defined
but PATH= has, then source will be saved to the path contained
in the string following PATH=.
This is the same as the CLI parameter '-p'.

SCREEN=Workbench
This will allow you to select which screen the source code will
use once compiled. Leaving this tooltype out will default the
source to finding the front most screen and using that. This
option can also be set in the GUI, before you convert any .gui
files.
As of V1.3 the ommision of this tooltype will default your source
to recreating the screen defined in the .gui file.
This is the same as the CLI parameter '-w'.

WINDOWLABEL=String ; new in 1.7
I have included this really for myself, but somebody else may
find it useful. Once you have created the source, your window
```

```
will have the value contained in String:
    i.e. WINDOWLABEL=MyWindow
will have the effect of the source creating the line:
    Window MyWindow,0,0.............
when you include this source into another program, your window
will have the value MyWindow instead of 0. If you omit this
tooltype then the source will give your window a value of zero.
This is the same as the CLI parameter '-wl'.

SCREENLABEL=String ; new in 1.7
This tooltype will act very similar to WINDOWLABEL accept that
it will substitute the screen number for the string.
This is the same as the CLI parameter '-sl'.

GLISTLABEL=String ; new in 1.7
This tooltype will act very similar to WINDOWLABEL accept that
it will substitute the gadgetlist number for the string.
This is the same as the CLI parameter '-gl'.

MENULABEL=String ; new in 1.7
This tooltype will act very similar to WINDOWLABEL accept that
it will substitute the menulist number for the string.
This is the same as the CLI parameter '-ml'.

TOGGLEGAD ; new in 1.7
This tooltype will add some code to the event handler that will
highlight the gadget selected by a keyboard shortcut as if it
was clicked on with the mouse.
This is the same as the CLI parameter '-t'.

ACTIVATESTRING ; new in 1.7
This tooltype works very similar to the above one, but will
place code into the keyboard reader so that presing the highlighted
letter of a 'String' or 'Integer' gadget will automatically activate
that gadget.
This is the same as the CLI parameter '-a'.

HIGHLIGHTLABEL ; new in 1.7
This tooltype will allow you to control how the program labels
appear in the source. By default, the labels are not highlighted,
but by using this tooltype will place a '.' in front of the labels
used in the source so that they will appear in the right hand
column of the Blitz editor.
This is the same as the CLI parameter '-h'.

DPEN=0 to 16
This will change the default DetailPen colour for your source window.
The number that follows the tooltype should be in the range that
corresponds with the amount of colours you are currently using.
The number is a palette reference not a 'DriPen' number. Colour
zero usually being the background colour (usually grey), working
up to the amount of colours in your current palette.
This is the same as the CLI parameter '-d'.

BPEN=0 to 16
This is the same as the DPEN tooltype, with the exception that it
sets the current BlockPen for your source window. See above for more
```

details on palette colours.
This is the same as the CLI parameter '-b'.

EXTENSION=String
This tooltype will allow you to change the default appended suffix.
Normally, BlitzGUIGen will save source files with the extension of
.BBGui, but the String following EXTENSION= will change this.
This is the same as the CLI parameter '-x'.

SAVEPATH=Path
This will now allow a different path to save the source files to. By
default, source files are saved to the same directory that the .gui
file was found in. By including a valid AmigaDOS path string after
SAVEPATH= you can override this setting.
This is the same as the CLI parameter '-v'.

MAKEXTRA
This switch will allow your source to have an associated .xtra
file. The main benefit of this is that when GTTags are
created for gadgets, the constant will be used instead of
the actual hex value. In the 'Compiler Options' menu, you will
see that 'Blitzlibs:Amigalibs.res' is already displayed in the
window. Also, 'Make Smallest Code' and 'Runtime Debugger' will
be switched on.
This is the same as the CLI parameter '-X'.

GTMENUS
This switch will allow you to create GadTools Menu's. This will need
the DefLibs file to have been re-created with the RIGTMenuLib.obj
file present in your userlibs drawer at the time of creation.
If you don't have this library, I suggest you contact Steve McNamara
of Leading Edge Software and enquire about collecting it. Enter
'GTMenus' as a tooltype into the Information window of the
icon and you're all set.
This is the same as the CLI parameter '-g'

NOKEYS
This switch will turn off creation of a loop to check for your
key presses. This is created automatically according to the defined
key shortcuts in your gadget text. The key immediately after the '_'
will be used as your key to check for. Case sense is switched to all
lowercase shortcuts. If you require upper case shortcuts, you'll have
to change them manually.
This is the same as the CLI parameter '-k'

NOMENUS
This switch will turn off creation of menu code. Any menus that have
been defined in the .gui file, will NOT be created. Useful, I
suppose, if you were composing an 'include' file where you only needed
some of the gui instead of all of it.
This is the same as the CLI parameter '-m'.

SELECT
This tooltype will turn on the creation of Select/Case loops
for gadgethit and key trapping. The main reason for including this is
because I have had problems with Gosub's from inside
Select/Case loops. When the return is executed, Blitz throws

up the 'Return Without Gosub' error. Using If/EndIf loops
cures this. Now you have the choice of which one you want to use. I
since found that the use of 'POP SELECT' before the Gosub cures this.
This is the same as the CLI parameter '-c'.

FUNCTIONS
This tooltype will now organise the source so that any sub-routines
will now appear at the top of the outputted code in the form of
'Statements' or 'Functions'. The calls to these sub-routines has also
been changed to reflect this. the default option here if this tooltype
is omitted is that sub-routines are created at the bottom of the source.
This is the same as the CLI parameter '-f'.

COMMENTS
This tooltype, if present, will cause somewhat detailed commenting of
the created source code. This can greatly simplify the understanding
of the code structure to the newcomer. For the more experienced
BlitzGUIGen user, leaving this tooltype out will supress the
embedded comments.
This is the same as the CLI parameter '-l'.

BACKFILL ; new in 1.8
This tooltype, if present, will cause BlitzGUIGen to create source
that contains code to create a backfill pattern. This pattern is the same
as seen in system requesters etc, and can be used for custom requesters
etc that you may create. This can be over-ridden by clicking the check-box
off in the main window.
This is the same as the CLI parameter '-B'.

FILLCOLOUR ; new in 1.8
This tooltype allows you to set the default colour for any backfill's you
may create. One colour will always be 0 (background) as the pattern created
is chequer-board style. The colour specified here will give a halftone
effect mixed with the default grey (0). eg: 'FILLCOLOUR=3' will make the
primary colour Blue.
This is the same as the CLI parameter '-C'.

1.12 Using parameters from the CLI

The following parameters are supported at the current time. These are:

-a This switch will place string activation code into
the keyboard reader loop. See the ACTIVATESTRING
tooltype for more information.

-b This key will allow you to change the default
BlockPen. See the BPEN= tooltype for more info.

-B This switch will enable the creation of backfill code.
See the BACKFILL tooltype for more information.

-c This switch will turn on the creation of Select/
Case loops for gadgethit and key trapping. See SELECT

tooltype for more information on this.

-C This parameter will allow you to set the default colour for the backfill pattern. If not present, it will default to 2 (white). eg: '-C 3' will make the primary colour 3 (Blue). See the FILLCOLOUR tooltype for more info.

-d This key will allow you to change the default DetailPen. See the DPEN= tooltype for more info.

-e This switch will turn off generating the main loop that handles IDCMP events. See the NOMAIN tooltype for more information.

-f This switch allows for the creation of Functions instead of sub-routines. See the FUNCTIONS tooltype for more information.

-g This switch will allow you to create GadTools menu's including nm_BarLabel. See the GTMENUS tooltype for more info.

-gl This key will allow you to change the 'GadgetList Label'. See the GLISTLABEL tooltype for more information.

-h This switch will enable label highlighting. See the HIGHLIGHTLABEL tooltype for more information.

-k This switch will suppress the creation of key press detection loop. See the NOKEYS tooltype for more info.

-l This switch will turn on the creation of embedded comments within the created code. By omitting this parameter, it will default to OFF (NO comments). See the COMMENTS tooltype for more info.

-m This switch will turn off generating the menu code that defines the menus. See the NOMENUS tooltype for more information.

-ml This key will allow you to change the 'Menu Label'. See the MENULABEL tooltype for more information.

-p This key allows you to enter the search path that BlitzGUIGen will look at for .gui files. The actual path must be a valid AmigaDOS path following the -p key. Refer to PATH= tooltype for more detailed info.

-s This switch will turn off generating the screen assignment for your window. See the NOSCREEN tooltype for more information.

-sl This key will allow you to change the 'Screen Label'. See the SCREENLABEL tooltype for more information.

-t This switch will provide gadget toggling code. See the TOGGLEGAD tooltype for more information.

```
-v      This key allows you to set the default path to
        save source files to. See the SAVEPATH= tooltype for
        more info.

-w      This switch will select the screen for your source
        to use. See the SCREEN= tooltype for more info.

-wl     This key will allow you to change the 'Window Label'.
        See the WINDOWLABEL tooltype for more information.

-x      This key will allow you to change the extension
        that is appended to the end of the source file. See the
        EXTENSION= tooltype for more info.

-X      This switch will allow you to create an ???.xtra
        file to accompany your source. See the  Using ToolTypes
        section for more detailed info on this.

Anything else typed as a parameter will be ignored and will invoke
the CLI template.

                        Refer to  ToolTypes
```

1.13 Contacting the Author

```
Should you feel it necessary to contact us for any reason,
the following addresses should be used, but it is advisable
to contact via EMail first to confirm address:

    Creative Software
    186 Shepcot House
    Cowper Gardens
    Southgate
    London
    England
    N14 4NT

or you can E-Mail if you prefer, to:

   simon@darkside.demon.co.uk
or bml@thenet.co.uk
or FidoNet Netmail at 2:254/524.28
or AmigaNet Netmail at 39.139/1.28
or the BLITZ_AMY echo on AmigaNet
or on the usenet BLITZ_LIST list.

Either method, none is preferred, although electronic mail will
probably gain a response quicker than Royal Mail.

Should you be unlucky enough to find any bugs in either the
executable or the created source, you should mail a copy of
the created  GadToolsBox  gui to the above address, in order
to remedy the problem.
```

Updates will be released to all Aminet mirrors once they
are finished, and new registrations will receive the latest complete
version in exchange for their fee.

1.14 Registration

Registration fees are as follows:

The cost of a keyfile is now 10 UKP. This can be sent to me as cash
or in the form of a cheque. Overseas users can opt to send a Euro-Cheque,
but however the method, the payment should be in Pounds Sterling. If any
other currency is used, it MUST include a further balance to allow me to
convert it into English Pounds. If it is found that there are insufficent
funds to cover the price of the keyfile, you will be notified by you're
preferred means, and the balance should be forwarded to me as soon as
possible. In the event that the balance is not recovered, you're keyfile
will NOT be sent to you, and the money will cover the incurred expenses.

Any subsequent updates and bug fixes will be released via the Aminet
archive, usually in dev/basic.

In all correspondence, please quote your full name and address,
and your current version and revision number. Also specify how you
would like to receive your keyfile, ie: by Snail Mail, EMail or by
Netmail. Electronic mail is probably the quickest way to get it, but
may not be the safest.

1.15 History of Releases

Below is an up to date list of releases and version
numbers, and some comments pertaining to each:

    Version 1.0

    Compile date: March 1995
    First official release. Put out to Beta-testers
    and messages posted to AmigaNet Blitz area to
    check out feedback. Uploaded to Waltons
    Mountain BBS, but not complete version. Minor bug
    found in #GTST_String and #GTTX_Text routines, not
    reading the default string properly. Hopefully
    fixed now!.

    Also added the owner routine for release. A crypted
    string embedded in executable, to allow for tracing

purposes. This allows me to track the coverage of
different sources of distribution.

Currently, version 1.0 does not support the creation
of custom screens, even though you may have created
one from inside GadToolsBox. This maybe
something to include in later releases, depending on
demand and support.
Also added the BlitzGUIGen.doc. This included for
people who don't like HyperGuide documents.

Version 1.01
Compile date: 14/4/1995
Added 2 more tooltypes:
EXTENSION= to allow the changing of default save suffix.
SAVEPATH= to redirect saving of source files.
Updated AmigaGuide + doc (again!).

Version 1.1
Compile date: 23/4/1995
After degrading back to 2.04, it was found after continuous
use, that there were still crashes just waiting to appear.
Small code rewrites and a re-compile later, and it seems stable
again.

Version 1.2
Compile date: 18/5/1995
Owing to a few complaints about supporting old software, I have
hopefully, allowed BlitzGUIGen to support GadToolsBox V2.0 b or
V2.0c. V2.0c still hasn't come into my possession due to the fact
that the 'V2.0c' distribution is actually V2.0b repacked!!!!
Unfortunately, this means it is, as yet, untested with V2.0c.

Version 1.2a
Compile date: 21/5/1995
This is an interim release as a minor (but inconvenient) bug
was found in the file verify routine. All fixed in this
version.

---------------Future Updates Only To Registered Users------------

Version 1.3
Compile date: 22/5/1995
Finally, I've supported screens created within GadToolsBox .
The type of screen is sensed and created appropriately.

The following screen types are:
Workbench - Blitz source uses Workbench Screen....Never! :-)
Public   - Blitz source uses FindScreen to use front most
            Public Screen.
Custom   - Blitz source creates screen as defined from GadToolsBox.
            Yet to support palettes, currently clones workbench
            palette.

Version 1.4
Compile Date: 28/5/1995
Well this version now supports palettes for custom screens.

The palette information stored in the .gui file will be converted into a set of RGB statements setting the screen colours to clone that as described in the GadToolsBox .gui file.

Version 1.4a
Compile Date: 12/6/1995
This upgrade thanks to Dave Dexter (tester). Font sensitivity now upgraded to set font in IntuiText. Any IntuiText printed into the window will now use the chosen font, instead of using the system's default.

The screen size reading routine also has a problem getting the width and height on some gui's, but I am unable to recreate this.

When no SAVEPATH= tooltype was specified, the source was being saved to launched path. Now corrected.

The routine for checking gui file version was rewritten, and now there should be no problems with un-supported GadToolsBox versions. Although, BlitzGUIGen is still to be tested with Version 2.0c.

----------Released into Public Domain again----------

Version 1.5
Compile Date: Under Development
The biggest feature of this release is the fact that it is now a key file upgrade. This version is freely distributable, but certain functions are unavailable without the key file. More on this later. Improvements and enhancements are:

Screen reading routine now corrected and reads sizes correctly.

The generated 'Main Event Handler' source now uses 'Select / End Select block, instead of If Then block. This makes the output easier to read.

A new tooltype has been added, 'MakeXTRA', and if present, this will create an ??.xtra file to accompany your source. See the Using ToolTypes section for more detailed info.

The tooltype reading code has been updated and cleaned, as there seemed to be problems, which were found during rewrite.

The version number and owner details are shown in the screen title when BlitzGUIGen is running.

The output source has been updated to include a GTArrowSize command to accompany the GTScroller gadgets. The arrow size wasn't being converted correctly, but this fixed it.

Also, support for PowerPacked .gui files has been added by request. This needs the DeCrunch executable in your C: drawer. An error requester will appear if it can not be found, but it will only look for it when needed IE: unpacking crunched gui's.

Unfortunately, the following functions are disabled without the

```
key file:
    Keyboard Shortcuts
    Loading Blitz From BlitzGUIGen
    Crunched gui support
    Creation of Custom Screens in source
    Creation of Menu Lists in source
    Creation of .xtra file
    Creation of Backfill code (1.8)

Version 1.6
Compile Date: 27/6/1995

Well here we go again.....

The most relevant update to this release is the support for
GadTools Menu's. This come courtesy of Steve McNamara of
Leading Edge Software. You will need a 'DefLibs' file created
with the RIGTMenuLib.obj file in your 'userlibs' drawer. This
will allow the use of the commands, added by this library, to
create GadTools Menu's. This option can be suppressed, and source
made to create standard Blitz menu's, by removing the 'GTMenus'
tooltype from the icon file.

Also, another tooltype has been added in order to suppress the
following function. The source can now create a loop to check for
key presses corresponding to your gadget shortcut's. This is done
completely automatically, but can be suppressed by adding the
tooltype 'NOKEYS'. See the  Using ToolTypes  section
for more details on these additions.

You may notice, or not as the case may be, that the tags supplied
for normal bevelbox's has been changed. This was because of 2
Enforcer hits during compiling. Strange error, but filling the
tags out completely fixed it.

Also the DetailPen and BlockPen defaults have been changed, as
GadTools menu's were corrupted by incorrect colours. A DPen
value of 0 and a BPen value of 1 looks correct, so I suggest
sticking with this. It also matches the DriPens settings set up
thro' Intuition.

Also I'm afraid I've junked the document file as well, to try
to keep the distribution size down. If you really need it, then
contact me and I'll pass it up.

One small note here is that a typing error was found in the CLI
parameter parser :( and rendered most parameters useless. Needless
to say, this has now been corrected.

----------------------BetaTesters Again------------------

Version 1.7a
Compile Date: 3/5/1996

This release now will handle multiple projects in one .gui file,
and will output a complete source for each window found in the file.
The filenames are differed by renaming each source file by inserting
```

the number of the window into the source filename.
If you elect to launch a new Blitz editor, BlitzGUIGen will inform you that multiple sources have been created and will show you the names of the source filenames. There will be a separate Blitz editor window opened for each source, and you can manipulate them as you need.

Also, the font sensitivity has been updated to supply the font to the screen as well. Therefore, if you are creating GadTools menus the font will be used for them as well. I am yet to work out an 'easy' way to get the screen title and window title to use the new font as well yet, but I'll keep trying. Otherwise it's starting to get more like a 'C' source creator :)

I have also added a few new tooltypes.
The first one is called 'SELECT', and this will allow us to specify using Select/Case loops for checking, instead of If/EndIf loops. The 'Select' loops seemed like a good idea to start off with, but I have run into a few problems with these loops. Mainly, executing a 'Gosub' from within the loop will throw up a 'Return Without Gosub' when trying to return. For this reason, I decided to go back to If/EndIf loops, but it's entirely up to you what you use.

Next is 'WINDOWLABEL'. All the xxxLABEL parameters have been included really for my personal use, but I thought that someone else may be able to make use of them as well. I find them invaluable, as they make my life considerably easier, but that's my programming style. 'WINDOWLABEL=windonum' will force the source to use 'windonum' as the value assigned to the window. All references to the window are now called via 'windonum'.
The same goes for 'SCREENLABEL','GLISTLABEL' and 'MENULABEL' which all apply the same as above.

The tooltype 'ACTIVATESTRING' has been added as well which will cause a small piece of code to be wedged into the keyboard reader so that key presses pertaining to 'String' or 'Integer' gadgets will be automatically activated.

Added 'HIGHLIGHTLABEL' for personal use really. This will force the source labels to be preceeded by a '.' making them appear in the right hand column of the Blitz editor.

The piece-de-resistance in this release is the tooltype 'TOGGLEGAD'. This will force the source to include a short routine at the bottom which will make the button react to a keyboard shortcut as though it had been clicked on with the mouse. By pressing the underscored letter on the button, the button will highlight, wait, and then unhighlight. This has been seen in program's such as Reqtools and SPOT, to name just a few. Now you can add this 'professional' look to your own code automatically.

Added yet another tooltype. 'FUNCTIONS' now allows you to have BlitzGUIGen to create 'Statements' out of the subroutines instead of having 'Gosub xxx'. This has been suggested by those of you that are heavily into structured (??) programs ;)
The default setting for this is OFF or to create sub-routines.

While on the subject of tooltypes, 'COMMENTS' now allows to you turn on the creation of embedded comments in the created source. By omitting it, this turns the option off.

Another bug squashed! :) It seems that GadToolsBox doesn't save a palette for the custom screen if you haven't editted it. Therefore the screen copies the WB palette, but this info is not in the .gui file. Consequently, BlitzGUIGen was trying to convert a palette that wasn't there :(That problem is sorted out as well now, and BlitzGUIGen will now sense if the palette is present, and if not, will clone the WB screen's palette.

Many rewrites of various code sections, formatting of output, and optimisation have taken the bulk of the time on this release. The main source improvement being the presence of proper menu checking loops in source now. Instead of just checking for any menu event and then quitting, we now have a full set of 'ItemHit' statements to build onto. Comments and labels are, of course, included.

Yet another improvement has taken place. I have now included the code to 'join' a listview to a string. If this option has been selected from within GadToolsBox, the source code generated will enable your code to be able to highlight the selected item from the listview in the string gadget placed underneath it without any coding on your behalf.

Finally ironed out an annoying bug that caused loads of 'hits' and a crash when exitting. Strangely enough, while rewriting some apparently unrelated code, it righted itself. Wierd!

Fixed another bug. My fault this one! BlitzGUIGen will look at the font used in the .gui file and see if it exists on the system. What I did was to scan the Fonts: dir to check, but the default font of 'topaz.font' size 8 isn't there, it's in the ROM. BlitzGUIGen threw up an error saying it couldn't find the font, and was changing the font to 'topaz.font 8' ;) Silly I know, but fixed now!

Also added the ability to assign the gadget ID's to a set of constants. This facility, according to my main feature suggester, makes the code more easily modified! Thanks Tony!

Crashing on exit returns <grrrr>. After labouring for ages to find it, Paul Juhasz steps in and saves the day. Hurrah!

--------------Version 1.7a Rev 37.140 finally goes public!---------------

Versions up to 37.143 contain small bug fixes and tidy-ups. Nothing major.

Version 1.8 Rev 37.145
Compile Date: 28/1/1997

The ouput source has been modified slightly to accommodate this versions improvements.
Mainly, this comprises of the ability to automatically create code to put a backfill pattern into the window. This pattern can be any one of the current screens palette by changing the first colour number.

(See commented code at this point...). The backfill will fill the entire window unless a bevelbox has been created, in which case, a rectfill on the boxes co-ordinates will clear the inside portion.

Also added a tooltype, 'BACKFILL', which will enable this feature permanently, although this setting is over-ridable by toggling the new checkbox in the main window. This facility is only available to registered users.

On reflection, the tooltype 'FILLCOLOUR' was added as well. This sets the colour for the backfill more permanantly. The default setting is 2 which will give a nice light grey effect, but you can change it on the fly by using the entry gadget located on the main gui.

Also added the code to make BlitzGUIGen open in the center of the screen. This is something that has been bugging me for a while, so I decided to add it now. And before you ask, 'No it's still not font sensitive!' ;)

Version 1.8 Rev 37.146
Compile Date: 5/2/1997

This release includes the expansion to the 'MAKETOGGLE' tooltype. If this option is specified, then any checkboxes that may have been created will now have toggle code automatically added to the key reading routine. This allows the checkbox to be alternated between it's states by pressing the corresponding key if a shortcut was created.

Version 1.8 Rev 37.148
Compile Date: 14/3/1997

Generaly messed around with the created source trying to organise it better, and make it clearer to understand.
Also moved the Backfill code into a Function/Subroutine, and replaced the BltPattern with Rectfill. This seems to be faster, but I'm sure you'll moan if I'm wrong. ;)
(This is a nice little routine that'll go well in anybody's library!)

Version 1.8 Rev 37.149
Compile Date: 29/5/1997

Cured a niggling little fault where some gadgets where being generated one pixel less on the height and a small fix for the default number in the 'GTInteger' gadget.
Also fixed border corruption in the DoBackFillMethod{} function.

Version 1.8 Rev 37.150
Compile Date: 6/6/1997

Decided it would be a good idea to create a GUI that used a Picasso screen mode. BlitzGUIGen fell over trying to create a 256 colour palette. This has now been fixed. Due to my nearsightedness, I only set the maximum colours to 32. Now you can handle up to 256 colours. Just in case, I have included a check, and if more colours have been defined, BlitzGUIGen will now knock it down to 256.

```
    Nevertheless, it appears that Blitz doesn't like the screenmode anyway, ↩
       but
    at least you should be safe with 256 colour native screen modes.
```

1.16 Oh Dear, I've Found A BUG!!!

```
    In the unlikely event you find a bug, you should mail me a copy
    of the saved .GUI file to one of the contact addresses given above.
    You should, also, never receive any error messages while creating
    source, but you may get an 'Unsupported Tag' error. If you do, please
    send me the .gui file and we can try and sort it out.
```

1.17 My little helpers!

```
    I would like to thank the following people for their assistance
    in testing and contributing to BlitzGUIGen. This list
    is by no means in any order.

        Paul Shandi     Tester and Feature suggester
        Dave Dexter     Tester and Error spotter
        Jaime Burns     Tester and Bug finder
        Perry Mowbray   Tester and Document formatting
        Phillip Eastham Tester
        Anthony Brice   Tester (V1.7a) and Feature suggester

        Mark Sibly  and
        Simon Armstrong for Blitz Basic (of course!)

        Steve McNamara of Leading Edge Software for his RIGTMenuLib.obj
        library, and the support for GadTools menu's it provides.

        The UK support center for finally getting my BUM8 to me,
        although I'd like to know what the hell happened to BUM 9!
        <later> Where's my BUM 10? ;)

        But most importantly, Paul Juhasz, without who's help 1.7a would
        never have got this far.

Thanks chaps.................
```

1.18 Things That Still Need To Be Done

```
Well there isn't much really left to implement, but you might
have some ideas on improvements.
If you want to see it, hassle me and send in your registrations,
that way I mat get some motivation!

Thanks to all of you in advance who will use BlitzGUIGen, and to
all who will register, thanks for your support.
```

```
See you all in a key file coming to you soon (I hope!).........
```