The GUI Builder Users Guide

The GUI Builder is a source-code generator that runs on your Mosaic touchscreen controller to assist you in creating an effective and interactive graphical user interface (GUI) for your application. While your controller's GUI Toolkit already provides the tools you need code a custom GUI for your application, the GUI Builder goes a step beyond that by coding your GUI for you. It gives you a what-you-see-is-what-you-get (WYSIWYG) approach to creating menus by placing and grouping graphics and buttons on your controller's display. You can edit and test your layouts directly on your target hardware. As you create your menus, you control the GUI Builder through a serial terminal while simultaneously viewing your menus on your controller's LCD/Touchscreen. When adding new graphics or buttons to menus, keys on the PC keyboard allow you to move them around to precisely where you want them. You can even use the touchscreen itself to drag and place the items with your finger. After creating your menu content, the GUI Builder then generates the source code that implements your user interface. You can use this source code directly in your application, or you can modify it as you like.

- ⇨ *The GUI Builder allows you to design your GUI visually, eliminating the need to painstakingly work out the screen coordinates of each of your objects.*
- ⇨ *All you need to get started using the GUI Builder is a graphics library. You may use our sample library or create your own custom graphics.*
- ⇨ *The GUI Builder generates all the code you need to define your objects using the GUI Toolkit. You may then further edit this code by hand if you wish.*

Creating Your Graphical User Interface

There are five steps to creating your custom GUI:

1. Rough-out a top-down design of your application's GUI;
2. Build a graphics library containing each of the graphics of your GUI;
3. Install the graphics library and the GUI Builder on your controller;
4. Use the GUI Builder to position your graphics, create your GUI's menus, and generate code; and,
5. Incorporate the generated code into your application.

These steps are discussed in the following sections of this document.

The accompanying GUI Builder Quickstart Guide document (the GUI_Builder-quickstart.pdf file in the distribution) contains a brief and informative overview describing how to install the software and exercise the built-in demonstration program.

Step 1. Rough-out a Design

The first step to designing your application's GUI is to do a rough pencil-and-paper design of its screen shots. Sketching out your user interface before you start programming will help you to think about how you want your application to be organized and how you want to present information to your user.

Each screen to be displayed by your application (what we'll call a menu) groups together graphic images and/or buttons. When the user presses a button your application may continue displaying the same screen, or remove it and display instead a new menu. To start your design ask yourself what information needs to be displayed at each step of your application's process, what decisions you'll expect of the user, and how those decisions affect the application's functioning. Then rough-out the screens by making simple paper-and-pencil sketches. At this stage you'll quickly realize where you may need additional menus, or where several menus should be combined into one.

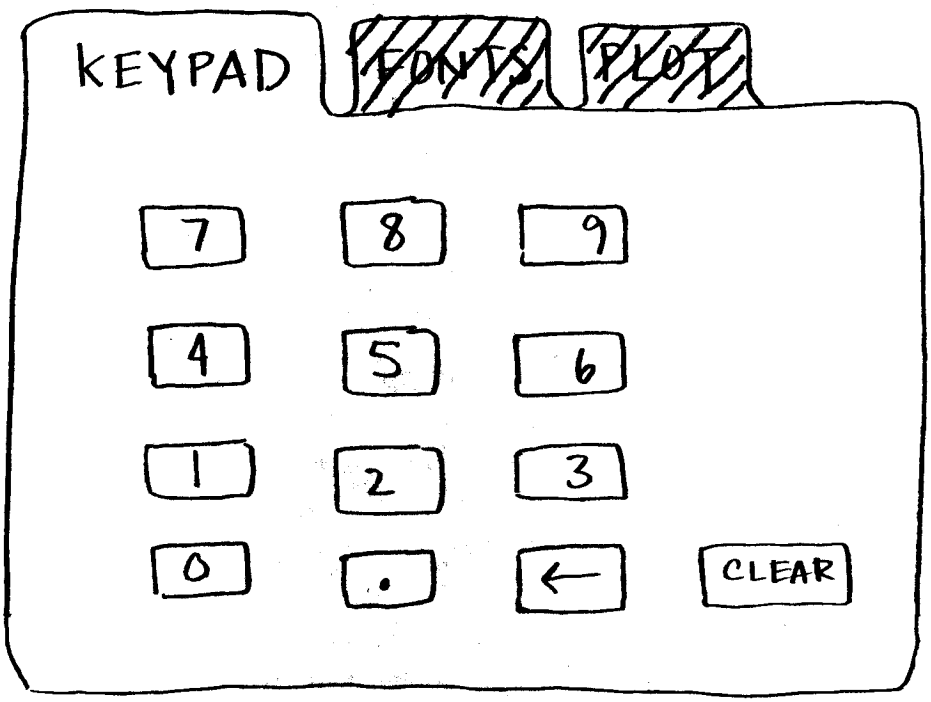

Figure 1-1 The sketch of the first screen of the demo program.

Figure 1-1 shows the type of rough sketch you might come up with. When you are satisfied with your sketches it's time to move on to the next step, designing any special graphics for your menus.

Step 2. Build a Graphics Library

The GUI Builder relies on a graphics library containing all of the graphic images your menus will use. These may include any special logos, illustrations, data plot elements, graphically rendered text, and button images. Once you've sketched out and designed your user interface, you need to create bitmapped images of your interface using an image editing program. The GUI Builder is provided with a default graphics library that contains a variety of button shapes and examples of other graphics.

Using the Default Graphics Library

To get you started there is a default graphics library included with your IDE that you may use as-is. You may also easily extend it to include your own custom images. Figure 1-2 shows the images included in the default library. You can see that there are two sizes of buttons and several different styles. Also, because each button image requires a "pressed" version for display while the user presses the button, there are solid, filled versions of each button type.

The Mosaic IDE (Integrated Development Environment) includes a Windows image converter tool for generating a graphics library from your own bitmap (BMP) or PaintBrush (PCX) files. The output of the image converter is a pair of files: an *image data file* called **`image_data.txt`**, and an *image headers file* called **`image_headers.h`** (or **`image_headers.4th`** for Forth programmers). The image data file is stored in the same directory as your images and when it is transferred to your controller, it stores your image data in RAM and then copies it into Flash. You must transfer the Image Data File using Mosaic's Terminal Program to your controller before you transfer your application code. When you use the GUI Builder you'll be able to download these files with the assistance of the command list items. Once your graphics are installed into the GUI Builder environment, you'll be ready to start creating your instrument's custom user interface.

If you wish to simply use the default graphics as is, you may do so by using the existing **`image_data.txt`** and **`image_headers.h`** (or **`image_headers.4th`**) files.

Creating Custom Graphics

While the GUI Toolkit comes with a default graphics library of useful buttons and images, you'll probably also want to use some custom images. The GUI Toolkit also includes a collection of interesting images for you to use. Take a look at them on your PC and copy those you wish to use to your own directory. Then add any additional images you wish to use for your buttons, screen labels, etc. Be sure to follow the guidelines described in the QScreen GUI Toolkit Manual regarding new images you create.

Images may be created using Microsoft Paint, which is included with all versions of Windows. You can also use other image editing programs such as Corel Draw, Photoshop, and Paint Shop Pro. The GUI Toolkit User's Manual contains instructions for creating and sizing graphic images. Briefly, you should create one graphic image for each indivisible graphic element on the screen, for example, for each button or picture that stands alone. Using the smallest images possible will allow you to save space in Flash memory since you don't want to store the white space between images. It

will also allow you to associate the images with individual buttons. The width of an image should be a multiple of 6 pixels so that the image may be positioned on a 6 pixel horizontal grid on a screen. Finer horizontal pixel placement and image widths would require bit shifting of each byte of the image every time it is drawn, significantly slowing performance of the GUI Toolkit. There are no limitations on the height of an image or the placement of an image in the vertical direction.

Compiling the Graphics Library

Once you have compiled a directory containing all of the images you intend to use in your user interface, the image converter compiles the image data into 2 files: the image headers file and the image data file. The image headers file is named **`image_headers.h`** (or **`image_headers.4th`** for Forth programmers) and the data file it generates is called **`image_data.txt`**. The image data file contains hexadecimal S-Records and certain instructions that install the data into the flash memory of the controller. The image data file typically only needs to be downloaded one time since it is stored in non-volatile flash. Flash memory is not cleared or easily corrupted even during a crash. The image headers file is much smaller and serves as a table of contents to the image data. This file simply associates each image's name with its actual location in flash memory. The GUI Builder stores this as part of the active session workspace in RAM. Thus, if the GUI Builder's session is reset, the GUI Builder will inform you that you need to reload the image headers file, even though the image data is still intact in the flash memory.

One concept that is very important is that you must always ensure that the image data you are using matches the image headers you are using. These files are always generated at the same time by the image converter. Whenever you decide to change or add to the image library, you must use the image converter to regenerate these files, and reload the image data as well as the image headers file. The image converter tool is also explained in detail in the QScreen GUI Toolkit Manual.

The good news is that there is no requirement to master these skills right away because we provide a ready-to-use sample image library with the GUI Builder in the **`\images`** directory. If you do not have this, contact Mosaic Industries and we will send you a copy free of charge. Below is an index of what the image library looks like.

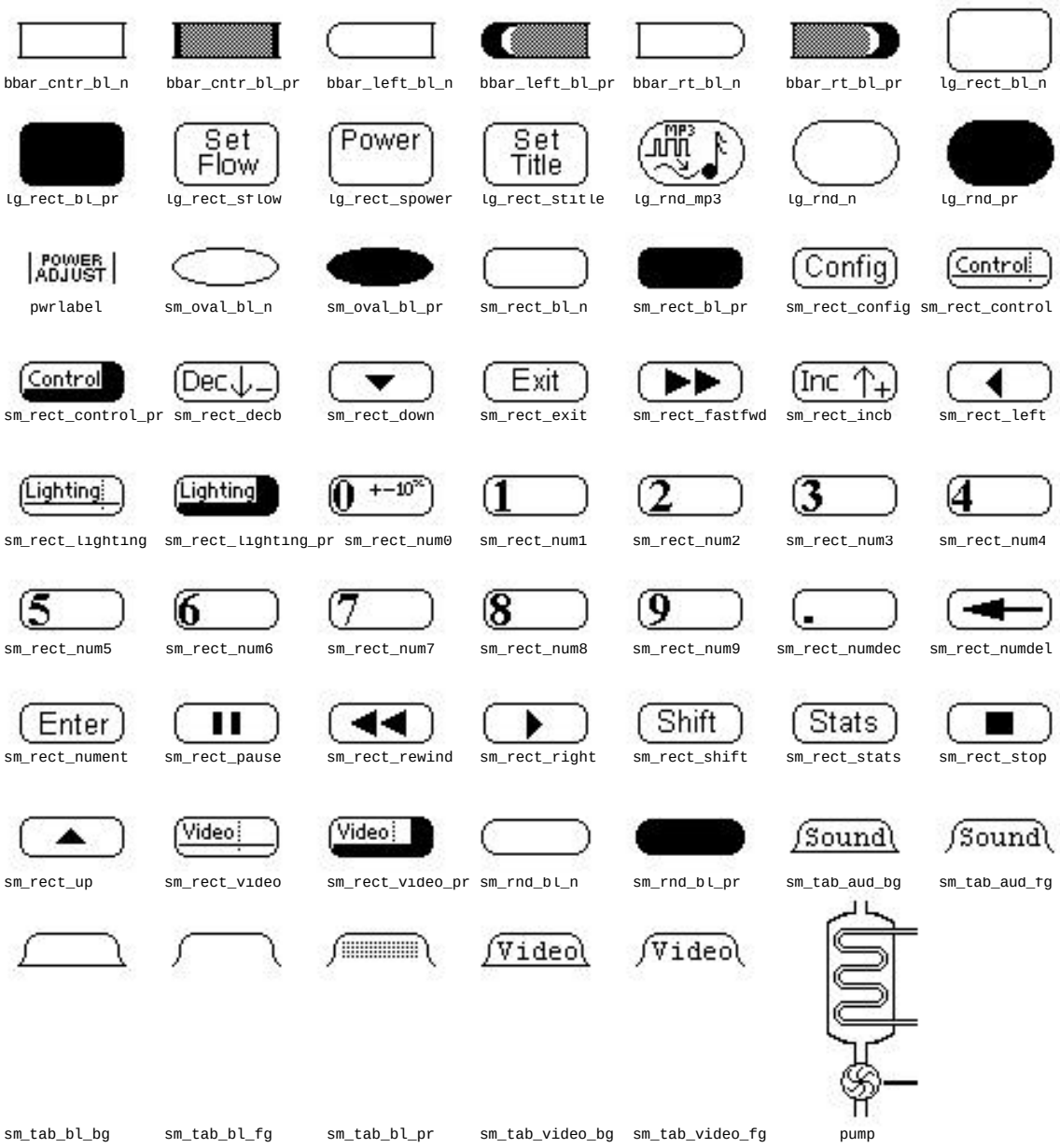

Figure 1-2 The default graphics library.

Step 3. Install the GUI Builder and Your Graphics Library

The GUI Builder is provided as a directory tree as shown in figure 1-3 that includes the default graphics library, GUI Builder software, documentation, and a pre-built GUI Toolkit kernel extension. We also provide an updated version of the image converter tool with the GUI Builder package.

Kernel extensions are modular software add-ons available through a web tool provided free of charge to Mosaic customers. Contact Mosaic Industries if you do not have login instructions for this tool. We have provided a pre-built GUI Toolkit kernel extension with the GUI Builder distribution, so you do not need to access the web tool to take advantage of the GUI Builder.

The GUI Builder will be packaged with future versions of the IDE (Integrated Development Environment, including the QED Terminal and TextPad editor). You must already have a working installation of the IDE before you begin this installation procedure. Contact Mosaic Industries if you need a replacement copy. The IDE is rooted in a directory called **\mosaic**, which is typically installed in the root directory of your C drive (**c:\mosaic**). The GUI Builder is shipped on a CD as a pair of directories that you simply drag and drop into your **\mosaic** directory. You can download the GUI Builder as a zip file which contains these two directories. Again, simply drag and drop them into your **mosaic** directory. They are called **GUI_Builder** and **Mosaic Image Converter**. The image converter directory simply contains a replacement executable for the image converter utility to ensure you are using the most up to date version (V1.10 or later). The **GUI_Builder** directory contains the default image library, example session, this document, and all the files needed to install the GUI Builder.

To install the GUI Builder on your computer, copy the **\gui_builder** directory structure into your **\mosaic** directory. This will overwrite your **\mosaic\Mosaic Image Converter** directory contents with the updated image converter utility.

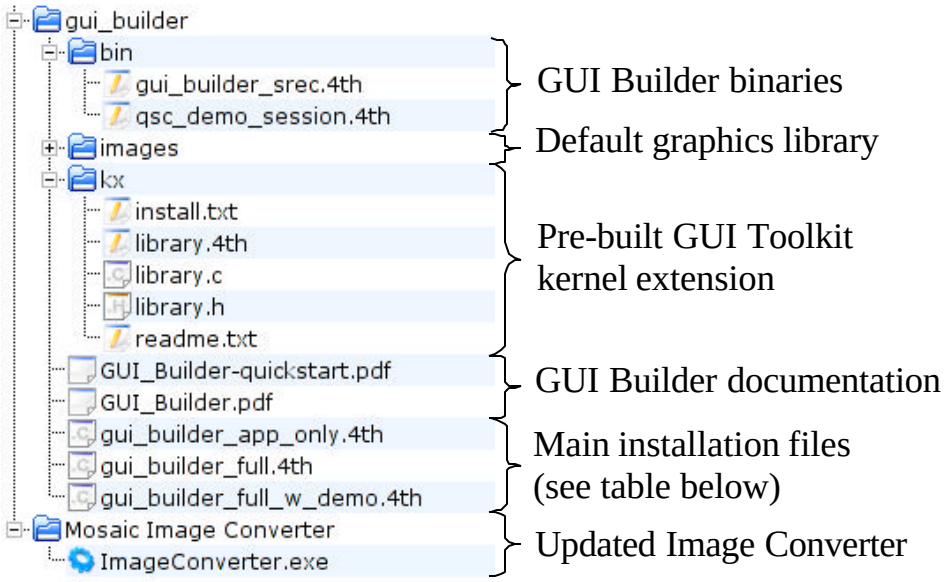

Figure 1-3 GUI Builder directory structure

Initially, you do not need to worry about any of the files in the **kx** or **bin** directories. Files from both directories are used by the main installation files. The **kx** directory contains an already built kernel extension of the GUI Toolkit which is used by the GUI Builder. The **bin** directory contains the GUI Builder install binaries. The default graphics library files and the processed image header and data files are stored in the **images** directory. Image header files have been generated for both C and Forth.

We provide 3 installation files that will install various components of the GUI Builder package. They are summarized in Table 1-1. Depending on which file you choose, you may install just the GUI Builder, the GUI Builder plus the GUI Toolkit kernel extension, or the aforementioned plus the GUI Builder demo session and graphics.

Start the QED Terminal application provided with your IDE, connect your PC's serial port to the controller, power up the controller, and verify communications by tapping the enter key. If you see 'ok' on your terminal screen, you can skip the following troubleshooting paragraph.

If you don't see 'ok' being printed on your terminal screen, verify that the controller is connected to the same serial port to which the terminal program is configured, and that the terminal is set to 19200 baud. If there is another auto-starting program running on the controller such as the GUI toolkit demo, you can clear it by performing the factory cleanup procedure: install the factory clean jumper and cycle power to the controller.

Once communications are established, choose 'Send File' from the 'File' menu of the terminal program, and select one of the main installation files as described in Table 1-1 from your **`\mosaic\gui_builder`** directory. ***If you are installing the GUI Toolkit for the first time, calibrate the touchscreen by choosing '`T. (Re)calibrate touch screen`' from the GUI Toolkit's command list and following the instructions.*** This step only has to be performed once after installing the GUI Toolkit kernel extension (which is installed as part of downloading **`gui_builder_w_demo.4th`** and **`gui_builder_full.4th`**). Now you're ready to use the GUI Builder.

> **Note: The GUI Builder runs best with 512k of RAM. If you have only 128k of RAM, you may have to save your session prior to reloading graphics, and then restore your session. For 512k units, click 'Advanced' in the Image Converter and set the RAM start address to 0x180000 (page 0x18). The default of 0x10000 (page 0x01) works on 128k units, but erases your session whenever you refresh graphics libraries. If this does happen, you can simply restore your session assuming you saved it.**

The GUI Builder starts running automatically upon boot once you have downloaded one of the files listed below. Moreover, it will automatically begin immediately following when the installation download completes. We provide three files that enable different ways of using the GUI Builder. They are summarized below:

Table 1-1 GUI Builder loader files and description.

Filename	Description
`gui_builder_app_only.4th`	This file installs the GUI Builder application only and configures it to autostart. The accompanying kernel extension, kx/install.txt must also already be installed. If you need to reinstall the GUI Builder but do not need to reinstall the kernel extension, this file will take less time to download. **Caution: if you install this and do not have the kernel extension installed, you may need to factory clean the controller in order to regain control of the board.**

Filename	Description
gui_builder_full.4th	This file installs the GUI Builder and the kernel extension. It will bring a virgin board to the point of autostarting the GUI Builder application. You will need to supply a graphics library once you start using the GUI Builder (you may use the prebuilt default graphics library stored in /images).
gui_builder_full_w_demo.4th	This file does everything gui_builder_full.txt does plus installs the default graphics library and a work session environment that includes several buttons and menus. If you are just getting started with the GUI Builder, install this file.

If you are just getting started with the GUI Builder, install **gui_builder_full_w_demo.4th**. This file will take about 4 minutes to download at 19200 baud, and it will fully equip your controller with a complete work environment that you can explore. When the download is complete, the GUI Builder should immediately start running. The GUI Builder should display a line like this:

```
Total Graphics: 62  Total Buttons: 18  Total Menus:  2
```

You can use the exercise menu function to view the two menus.

Don't forget to calibrate the touchscreen if this is the first time you have installed the GUI Builder.

Play around with this demo session, add and remove buttons from the menus, and get to know this tool. More information about the demo session environment can be found in the GUI Builder Quickstart Guide included with this distribution. Reading the Quickstart Guide and using the demonstration program will help you to quickly learn about the interactive capabilities of the GUI Builder.

When you are ready to begin work on your own project, clear the demo session by selecting '`F. Fully reset the GUI Builder (erasing any current work)`' and then install whatever graphics library and symbols file you intend to use. If you want to continue working the default graphics library after using the demo, then you needn't reinstall the image data – just the image headers. After resetting the session, you must always reload the graphic symbols using option '`A. Load graphics symbols (image_headers.h/4th - NOT image_data.txt)`.' Remember that you must install the image data itself using the option '`G. Install new Graphics library data (image_data.txt)`' any time you change the graphics library using the Image Converter.

If you wish to restore the demo session, simply choose option '`B. Load prior session`' and download the **qsc_demo_session.4th** file from the **\bin** directory. You do not need to install the default graphics library's image data file unless you have been using a different graphics library.

Step 4. Create Your Screens and Menus

Once you have decided on a preliminary set of graphics and loaded the graphics library and the graphics symbols into the GUI Builder, it's time to start realizing your roughed out user interface. Remember that the graphics alone do not have any button-like behavior (such as being sensitive to being pressed on the screen).

You should first create some buttons so that you'll have fodder for your first menu. When you select the option to create a button (described in detail below), you will be asked to specify at the very least a *draw graphic* and a *press graphic*. These refer respectively to the graphic that represents a button that has just been placed on the screen, and the graphic representing the button when it is being pressed. The third graphic that describes a button is the *release graphic*. In many applications, it is set to be the same as the draw graphic. The GUI Builder automatically sets the release graphic to be the same as the draw graphic, allowing you to change it later if you wish.

After you have some buttons, create your first menu. Add any static graphics such as logos and then place some of your buttons on the screen. You can choose to 'exercise the menu' which displays the menu with active buttons that you can press to simulate the operation of the menu. You can always return to edit a menu, even if you have created other buttons or menus in the mean time. Note that buttons are objects that are referenced by menus, not copied into them. You may reuse the same button in multiple menus, and any changes you make to the button are evident in all places where the button is used.

You can save your current work at any time using the Save Session feature of the GUI Builder. You will be prompted to invoke the "Save File" item from the File menu of the terminal. After you confirm that this has been done, all the data needed to resume from your current point of work will be sent as a text stream to the terminal and stored in the file that you designated. You can use the terminal to send this file to the controller at a later time to restore the session. This feature is discussed in detail in the "Save this session" item in the section titled "The Main Command List" below.

Step 5. Incorporate the GUI Code into Your Application

Once you are ready to try out the actual application you've been building, it is time to generate the source code. The GUI Builder can generate the source code for a session at any time using '`D. Generate GUI source code`' from the main command list. The generated source code references a few files needed for it to compile. Depending on where you save your source file in relationship to the kernel extension and image files, you may need to change the **`#include`** statements in the generated source code. C programmers should use the C compiler included with the IDE to compile the source code, creating a *.dlf download file. Forth programmers will directly send the source code file to the controller.

To test your source code, you will need to remove the GUI Builder from memory using the '`!. Remove the GUI Builder from your touchscreen controller`' option on the GUI Toolkit's main command list. This will leave the controller ready to accept either the Forth source code file or the compiled C source code file. The GUI Toolkit and graphics data are still loaded into the controller's flash, so they do not need to be reloaded. You can retain the simple kernel extension file that was loaded with the GUI Builder, or you can load and use a different set of kernel extension files to include needed drivers for Wildcards or other hardware. When you download the generated GUI code, type '**`main`**' at the Forth prompt to start the program. It will show a list of your menus and you will be able to choose which one to view.

Once you are satisfied with this skeleton code, then you are ready to integrate it into your application program. Define and post handlers for each button to replace the simple print statement handlers in the skeleton code, and add any computation and control capability that your application requires.

Using the GUI Builder

When you have finished downloading any of the installation files as described in Table 1-1 above, the GUI Builder starts automatically. It displays its main command list in the terminal window. You communicate with it through the controller's serial port to interactively compose your graphical user interface. When you are done, the GUI Builder generates source code for you to incorporate into your application code. You then load your application code into the controller for testing and further development.

The Main Command List

When you first start the GUI Builder, you see a shortened version of the main command list in the terminal window on your PC. This is because there are not yet any GUI Toolkit objects. Let's start by loading some graphics. The main command list looks like this prior to loading any graphics:

```
Total Graphics:  0  Total Buttons:  0  Total Menus:  0


It looks like you don't yet have any graphics available.  The first step is to
load some.  Once you have loaded graphics, then it is possible to create buttons
and menus.

+General Options:
  A. Load graphics symbols (image_headers.h/4th - NOT image_data.txt)
+Session Management:
  B. Load prior session
  C. Save this session
  D. Generate GUI source code
+System:
  E. Fully reset the GUI Builder (erasing any current work)
  F. Set preferences
  G. Install new Graphics library data (image_data.txt)
  T. (Re)calibrate touch screen
  !. Remove the GUI Builder from your touchscreen controller
  Q. Exit to a forth prompt (The Autostart will remain in effect)
*************************
Choose an item by typing its number/letter:
```

In all parts of the GUI Builder, the Esc (Escape) key exits the current prompt and returns to the previous command list level. In the main command list however, the Esc key is ignored. Many options are disabled until you provide some graphics.

Start with a Graphics Library

After you install the GUI Builder application, it immediately begins executing. If you installed the file **gui_builder_full_w_demo.4th** then a graphics library is already available, along with a valid work session including some buttons and GUI menus. If you used one of the other GUI Builder installation files, you must load a graphics library. The command lists indicate that there are no graphics until you install a graphics library, and many options are not available until you have completed this step.

Go ahead and install a graphics library as discussed in the previous sections. If you have not yet installed your graphics library's image data file select option `G. Install new Graphics library data (image_data.txt)` and follow the instructions to download your image library. Then, select '`A. Load graphics symbols (image_headers.h/4th - NOT image_data.txt)`' and you will then be prompted for the `image_headers.h` (or `image_headers.4th`) file. Now, the main command list should look more like this:

```
Total Graphics: 62  Total Buttons:  0  Total Menus:  0


**********Main***********
+Menu operations (you have not yet created any menus):
  1. Create a new menu
  2. Edit an existing menu      [disabled]
  3. Delete an existing menu    [disabled]
  4. Exercise an existing menu [disabled]
+Button operations (you have not yet created any buttons):
  5. Create a button
  6. Edit a button              [disabled]
  7. Delete a button            [disabled]
  8. Exercise a button          [disabled]
  9. Choose button template
+General Options:
  A. Load graphics symbols (image_headers.h/4th - NOT image_data.txt)
+Session Management:
  B. Load prior session
  C. Save this session
  D. Generate GUI source code
+System:
  E. Fully reset the GUI Builder (erasing any current work)
  F. Set preferences
  G. Install new Graphics library data (image_data.txt)
  T. (Re)calibrate touch screen
  !. Remove the GUI Builder from your touchscreen controller
  Q. Exit to a forth prompt (The Autostart will remain in effect)
**************************
Choose an item by typing its number/letter:
```

This is the full main command list. Many of the options are marked disabled because they require that you have one or more objects of that type to be useful. Graphics are the basis for creating other types of objects (such as buttons and menus).

Warning about Changing Graphics Libraries

It is possible that you will discover the need to modify or add graphics during a work session. This can be done if care is used. As long as pages **0x01**, **0x02**, or **0x03** of RAM are not disturbed, the session should remain intact. By default, the Image Converter uses RAM pages starting at **0x18** and places the graphics data at flash page **0x10**, so pages 1, 2 and 3 are not disturbed by the graphics update. Nevertheless, if you have invested a lot of time into your session, it is wise to save it using the '`C. Save this session`' function before attempting to update the graphics library. After using the Image Converter program to create the updated library, use the '`G. Install new Graphics library data (image data.txt)`' option as discussed in the previous section. Then use the '`A. Load graphics symbols (image headers.h/4th - NOT image data.txt)`' function to load the image headers file.

Warning #1: Do not attempt to exercise the menu or edit any GUI objects until you have also loaded the graphics symbols file. Using an out-of-sync symbols file will likely cause a crash.

Warning #2: If you have loaded a previously saved session that was saved with a different graphics library, be sure to reload the graphics symbols for your current graphics library after restoring the saved session.

Warning #3: Although there is some support for the handling of missing graphics, we strongly advise against removing graphics from the library without also removing references to them in any menus or buttons. If you plan to remove or rename a graphic from the library, make sure you aren't still using it in any of your objects.

The GUI Builder Command List Reference

This section describes the operation of each of the GUI Builder command lists. Listing 1-1 illustrates the hierarchy of command lists. They are shown as three listings, ***main command list***, ***menu edit command list***, and ***button edit command list***. A brief description of each command list option is provided in the listings. A glossary of command list items and their functions is then presented.

Listing 1-1 GUI Builder Main Command List

```
**********Main***********
+Menu operations:
  1. Create a new menu
```

Queries for a new menu name

Goes to *menu edit command list*

2. Edit an existing menu

Shows a list of current menus, user selects one

Goes to *menu edit command list*

3. Delete an existing menu

Shows a list of current menus, user selects one, confirms, deletes

4. Exercise an existing menu

Shows a list of current menus, user selects one, menu is demonstrated

+Button operations:

5. Create a button

Queries for a new button name

Goes to *button edit command list*

6. Edit a button

Shows a list of current buttons, user selects one

Goes to *button edit command list*

7. Delete a button

Shows a list of current buttons, user selects one, confirms, deletes

8. Exercise a button

Shows a list of current buttons, user selects one, button is demonstrated

9. Choose button template

Shows a list of current buttons, user selects one, future buttons inherit the properties of the selected button

+General Options:

A. Load graphics symbols (image_headers.h/4th - NOT image_data.txt)

Prompts user to download image headers file

+Session Management:

B. Load prior session

Prompts user to download a saved session file to the GUI Builder

C. Save this session

Prompts user to capture a session file from the GUI Builder

D. Generate GUI source code

Prompts user to specify language (C or Forth) and to prepare to capture the output of the GUI Builder

```
+System:
  E. Fully reset the GUI Builder (erasing any current work)
```

Prompts user to confirm, then resets session

```
  F. Set preferences
```

Goes to the *preferences command list*

```
  G. Install new Graphics library data (image_data.txt)
```

Prompts user to download image data file

```
  T. (Re)calibrate touch screen
```

Displays calibration instructions on touchscreen requiring the user to touch each of three points

```
  !. Remove the GUI Builder from your touchscreen controller
```

Prompts user to confirm, then removes the autostart vector and calls 'cold'

```
  Q. Exit to a forth prompt (The Autostart will remain in effect)
*************************
```

Listing 1-2 GUI Builder Menu Edit Command List

```
Currently,15 button object(s) and 1 graphic object(s) in this menu
*******-Edit this Menu*-******
1. Add a graphic object to this menu
```

Shows a list of graphics with the option to browse, user selects one to add, then user interactively positions the graphic on the screen

```
2. Add a button object to this menu
```

Shows a list of buttons with the option to browse, user selects one to add, then user interactively positions the button on the screen

```
3. List the objects in this menu
```

Shows a list of all objects currently in the menu

```
4. Move an object in this menu
```

Shows a list of objects in the menu with the option to browse, user selects one, then user interactively repositions the object on the screen

```
5. Delete an object from this menu
```

Shows a list of objects in the menu with the option to browse, user selects one, confirms, then item is deleted from the menu

```
6. Exercise this menu
```

Draws the menu onto the screen with working buttons for user to test

```
****************************
```

Listing 1-3 GUI Builder Button Edit Command List

```
************Edit this Button***********
1. Specify this button's DRAW graphic.
```

Shows a list of graphics with the option to browse, user selects one, it becomes the button's draw graphic. If release is not set, this sets release as well to the user's selection

```
2. Specify this button's PRESS graphic.
```

Shows a list of graphics with the option to browse, user selects one, it becomes the button's press graphic

```
3. Specify this button's RELEASE graphic.
```

Shows a list of graphics with the option to browse, user selects one, it becomes the button's release graphic

```
4. Specify this button's behavior parameters.
```

Shows a list of button flags and their current settings, user can toggle them on and off

```
5. Specify this button's 4 internal text strings (default = blank)
```

Shows a list all four lines of text a button can display, user can edit any of them

```
6. Exercise this button.
```

Demonstrates the button on the touchscreen

```
****************************************
```

The Main Command List

```
1. Create a new menu
```

This option allows you to create a new GUI Toolkit menu. When you select this option, you will be asked to enter a name for your new menu. You may use upper or lower case in the menu name, but it will be forced to lower case in the generated code. The GUI Builder then enters the GUI menu editing command list for the GUI menu item you've just created. Within that command list, you will be able to add objects to the GUI menu and adjust their positions on the screen. GUI menus are simply containers for graphic objects and button objects. Generally, GUI menus aren't useful unless they have buttons in them as well. Although it does not matter in what order you create your objects, it generally makes sense to create buttons first then create menus in which to put them. In practice, you may create some buttons and menus, then create additional buttons as your design evolves.

```
2. Edit an existing menu
```

This option displays an enumerated list of menus and prompts you to choose one to edit. Once you select a menu to edit, the menu editing command list will appear in your serial terminal. From this

command list, you can add or delete objects and adjust their positions within the menu. See below for more details.

`3. Delete an existing menu`

This option displays an enumerated list of menus and prompts you to choose one for deletion. When a menu is deleted, the objects it contains are not deleted. You can still use the buttons and graphics in other menus. There is no restriction on your ability to use the same graphics and buttons in multiple menus.

`4. Exercise an existing menu`

This option displays an enumerated list of menus and prompts you to choose one to exercise. Your selection then appears on the LCD screen. When you exercise a menu, all of its objects are drawn to the LCD just as they would be in your own application. The buttons work when you press them so you can get a feel for how the menu you've made will behave in your final application. Note that the *activate on press*, *activate on release*, and *beep* settings are not evident when you exercise a menu within the GUI Builder, but these settings will be visible when the generated code is loaded and tested. To exit the exercise menu function, simply press the Esc or enter key.

`5. Create a button`

This option allows you to create a new button. Button objects are defined in terms of the graphic image to be used when it is first drawn to the screen (the *draw graphic*), the image shown while the operator presses it (the *press graphic*), and the image drawn when the operator releases it (the *release graphic*). When you create a new button, you must first supply a unique non-case-sensitive name for the button. Then, the GUI Builder will display the button edit command list for your new button (assuming the name you supplied was accepted). Within this command list, you will be able to adjust the button's appearance and behavior as well as test an instance of the button on the touchscreen. Note that the location on the screen a button will ultimately occupy is assigned when the button is added to a menu; this placement information is not part of the button itself.

`6. Edit a button`

This option displays a list of current buttons. After selecting a valid button from the list, the GUI Builder displays the button edit command list on your PC terminal. In the button edit command list, you may assign graphic images to the button's states (first drawn, pressed, released), set the button's behavior, test the button on the screen, and assign any text labels to be displayed with the button.

`7. Delete a button`

This option displays a list of current buttons. You may then select a button to delete. After confirming, the button and all instances of it in all menus will be deleted from your GUI Builder session. This action does not delete the graphics that may have been associated with the button.

`8. Exercise a button`

This option allows you to choose a button to be displayed by itself on the LCD as a live GUI object. You should be able to test the button by touching it on the touchscreen.

`9. Choose button template`

When creating many buttons, it is sometimes handy to have all buttons automatically inherit the defaults of one you've already defined. This option allows you to select an existing button whose characteristics will become the initial defaults for new buttons you create. If you do not specify a

button template, there is no inheritance by default. You can also use the F. Set preferences option to specify that each new button will inherit the characteristics of the last button defined.

A. Load graphics symbols (image_headers.h/4th - NOT image_data.txt)

This option allows you to download the graphics image header file to the controller. When you select this option, you will be instructed to download the image headers file corresponding to the graphics library that is currently loaded into your controller. If you need to load your graphics data, see 'G. Install new Graphics library data (image_data.txt)' below.

B. Load prior session

Loads a prior saved session. You can create a save session file based on the current environment using 'C. Save this session,' described below. When you choose this option, you will be prompted to initiate a capture of the serial terminal by selecting 'File'→'Send File' from the terminal program's menu bar. Select the saved session file that you created with the option, 'C. Save this session,' described below. At the end of the download, the main command list should reappear.

C. Save this session

This option provides a means of saving a snapshot of a GUI Builder work session as a file on your PC. The save session option prompts you to begin a file transfer from the touchscreen controller to the PC, and then the GUI Builder dumps the data needed to restore the current session. When you use this function, you will be prompted select 'Save File' from the 'File' menu in the terminal program window to start saving the session file. When the transfer is complete, be sure to stop the capture by selecting 'Close File' from the terminal's 'File' menu before hitting any key to return to the main command list. Using the 'B. Load prior session' command, you can then download this saved file to the controller to duplicate the previous environment. Note that the graphics symbols are also part of a session environment. After restoring a session, there is no need to reload the image headers unless you have changed the graphics library.

D. Generate GUI source code

This option first asks you if you are using C or Forth, then prompts you to set your terminal to capture the data that will follow by selecting 'File'→'Save File' from the terminal program's menu bar. This text comprises a complete source code file to define and test your newly designed user interface. Dummy routines containing simple print statements are the button handlers that you will later replace with your own handler code. The generated source code also includes a simple 'main' routine that allows you to select what menu to display on the touchscreen. See the section above titled "Step 5. Incorporate the GUI Code into Your Application" for more information.

E. Fully reset the GUI Builder (erasing any current work)

This option performs a session reset. It deletes all objects from the GUI Builder environment and resets all the defaults. This action is automatically performed on the first startup or on a restart if the RAM has been corrupted.

F. Set preferences

This option displays the preferences edit command list. In that command list, you can fine tune certain aspects of the GUI Builder's behavior such as button templates, command list verbosity, etc. The default preference list is as follows:

Index Preferences ..... Current State

```
  1.  Continuous menu redraw while placing items          ON
  2.  Use last button created as template for next button  OFF
  3.  Draw grid on display                                 ON
```

The first option controls the redrawing of other menu items while placing new ones. If this is turned off, dragging a new item across an old one will erase it in the path of movement. This is much faster than having the object be redrawn each time, but it doesn't look as nice. In practice, it should not be necessary to turn this option off.

The second option allows you to automatically have the last drawn button act as the template for the next one. This option is automatically disabled if you set the template to a specific button on the main command list.

The third option enables or disables the placement grid when positioning menu objects. We find it helpful, but if you would prefer not to see the grid, simply disable it here.

```
T. (Re)calibrate touch screen
```

The QScreen controller uses an analog touchscreen which must be calibrated. This calibration must be performed once after installing the GUI Builder. You will also need to include this functionality in your end application. See the GUI Toolkit documentation for more information regarding touch-screen calibration.

```
!. Remove the GUI Builder from your touchscreen controller
```

This option un-installs the GUI Builder from your system by removing the autostart vector and performing a cold restart. After that, your controller is ready for your code. If your application needs a different set of kernel extensions to support Wildcards or other features, remember to install the proper set before loading your application. If you are just testing the generated source code, you can use the prebuilt kernel extension provided with the GUI Builder.

```
Q. Quit the GUI Builder program
```

This option is simply terminates the GUI Builder and runs the Forth monitor. Since the autostart remains in effect, the GUI Builder will restart after any reset or if the Forth monitor experiences an error (causing it to call QUIT). This option is generally only useful to experts and should never be needed during normal use of the GUI Builder.

The Menu Edit Command List

The menu edit command list always shows the status of your menu in terms of the number of object types you currently have in your menu:

```
Currently,15 button object(s) and 1 graphic object(s) in this menu
```

```
*******Edit this Menu********
1. Add a graphic object to this menu
```

This option shows you a list of available graphics and allows you to browse the list interactively on the screen or choose one by number directly from the list. After you select a graphic, it will be

added to the menu in the upper left corner of the screen. You can move the graphic around on the menu with the keyboard, or by touching any location on the screen, which causes the upper left corner of the graphic to be placed at that screen location. You can also drag your finger around on the screen to locate the graphic. Hit enter to place the graphic at the displayed location or Esc to abort the placement and leave the graphic in the upper left corner of the screen.

```
2. Add a button object to this menu
```

This option shows you a list of available buttons and allows you to browse the list interactively on the screen or choose one by number directly from the list. After you select a button, it will be added to the menu in the upper left corner of the screen. You can move the button around on the menu with the keyboard, or by touching any location on the screen, which causes the upper left corner of the button to be placed at that screen location. You can also drag your finger around on the screen to locate the button. Hit enter to place the button at the displayed location or Esc to abort the placement and leave the button in the upper left corner of the screen.

```
3. List the objects in this menu
```

This option simply prints a listing of the objects currently in the menu.

```
4. Move an object in this menu
```

This option shows you a list of objects currently in the menu and allows you to browse the list interactively on the screen or choose one by number directly from the list. After you select one, you may reposition it using the keys on the keyboard as instructed, or by touching any location on the screen, which causes the upper left corner of the object to be placed at that screen location. You can also drag your finger around on the screen to locate the object. Hit enter to place the object at the displayed location or Esc to abort the placement and leave the object in its previous position on the screen.

```
5. Delete an object from this menu
```

This option shows you a list of objects currently in the menu and allows you to browse the list interactively on the screen or choose one by number directly from the list. After you select one, you will be asked to confirm the deletion, and then the object will be deleted from the menu.

```
6. Exercise this menu
```

This option draws the menu on the screen and fully simulates the live GUI. The buttons will work if you press them, and you can get a feel for your user interface. Note that buttons don't beep even if the beep flag is set. The beep flag will govern the behavior of the buttons in the source code you generate.

```
****************************
```

The Button Edit Command List

```
*************Edit this Button************
1. Specify this button's DRAW graphic.
```

This option shows a list of all available graphics and allows you to browse the list interactively or select one directly by its number in the list. The graphic you select becomes the button's draw graphic, that is, the graphic shown on the screen when that button is drawn. If the release graphic is not set, then it is also set to the graphic you selected. For typical buttons, the release graphic matches the press graphic.

```
2. Specify this button's PRESS graphic.
```

This option shows a list of all available graphics and allows you to browse the list interactively or select one directly by its number in the list. The graphic that you select becomes the button's press graphic, which is the graphic drawn when the user presses that area on the touchscreen.

```
3. Specify this button's RELEASE graphic.
```

This option shows a list of all available graphics and allows you to browse the list interactively or select one directly by its number in the list. The graphic you select becomes the button's release graphic, that is, the graphic drawn when the user removes their finger from the touchscreen after pressing the button. Typically, this is the same as the draw graphic.

```
4. Specify this button's behavior parameters.
```

This option shows a list of toggle flags that control a button's behavior. See the GUI Toolkit documentation for a detailed explanation of the flags.

```
5. Specify this button's 4 internal text strings (default = blank)
```

This option takes you to a small list showing each of 4 lines of text your button may contain. You may select each line individually to edit. Note that, when placed on the screen, the text is drawn at a granularity of 6 pixels wide by 8 pixels tall per character. When you move buttons on a menu to place them, you will see this effect.

```
6. Exercise this button.
```

This option draws the button in the upper left corner of the screen and allows you to test it. When you press that area of the touchscreen, its draw graphic will appear, and when you release it, the release graphic will appear.

Thanks for Using the GUI Builder

At this point, the GUI Builder is a beta version tool. We would like your input regarding various ways to improve it. If you have any questions about the GUI Builder or bugs to report, please contact us by sending email to gui_builder@mosaic-industries.com. We can also be reached by phone at (510) 790-1255.