ThreadGUI Doc

Joshua Ashby

Linux And Sci
`http://joshashby.com`
`joshuaashby@joshashby.com`

December 29, 2009

Contents

1 Introduction **2**

2 Setting up **2**

3 testApp.cpp/.h **2**

4 threadedObject.h **2**

4.1 buttonAction(int button); 2
4.2 threadedObject(); 3
4.3 start(); 3
4.4 stop(); 3
4.5 threadedFunction(); 3
4.6 label(int x, int y, char str[20]); 3
4.7 labelGreen(int x, int y, char str[20]); 3
4.8 labelGrey(int x, int y, char str[20]); 3
4.9 rectButton(int x, int y, char srt[15], int color); 3
4.10 rectButtonGrey(int x, int y, char srt[15], int color); 4
4.11 rectButtonGreen(int x, int y, char srt[15], int color); 4
4.12 roundButton(int x, int y, int h, char srt[15], int color); 4
4.13 roundButtonGrey(int x, int y, int h, char srt[15], int color); 4
4.14 roundButtonGreen(int x, int y, int h, char srt[15], int color); 4
4.15 click(int x, int y); 5
4.16 click(int color); 5
4.17 line(int x,int y, int xc, int yc, int thick); 5
4.18 lineGreen(int x,int y, int xc, int yc, int thick); 5
4.19 lineGrey(int x,int y, int xc, int yc, int thick); 5
4.20 lineBlue(int x,int y, int xc, int yc, int thick); 5
4.21 bullet(int x, int y, char str[20]); 5
4.22 bulletGreen(int x, int y, char str[20]); 7
4.23 bulletBlue(int x, int y, char str[20]); 7
4.24 mlBullet(int x, int y, char str[20]); 7
4.25 mlBulletGreen(int x, int y, char str[20]); 7
4.26 mlBulletBlue(int x, int y, char str[20]); 7
4.27 text(int x, int y, float data); 7

Abstract

ThreadGUI is a GUI system that is made to be easy to use and add on to. It builds the GUI in a second thread so the GUI does not have to mess with timing or sleep functions in the main thread and so on. It includes square buttons, circlular buttons, labels, bullet list items (single and dual line), lines, and single line text.

1 Introduction

Writing ThreadGUI came to mind when I was working on programming a new GUI for my robot BOB. I had recently started to program a few projects in C++ again, and was getting the hang of using the Open Frameworks library. Seeing that I may want to use Open Frameworks, and the lack of GUI systems for it, I decided to write one my self.

2 Setting up

ThreadGUI uses the Open Frameworks C++ libraries, and comes packed with only the project files. as a result you'll have to download Open Frameworks FAT. after downloading and setting up, download all the files from the master branch of ThreadGUI from here:
git clone git://github.com/JoshAshby/ThreadGui.git
Place (or clone) these files into a file in the Open Frameworks apps/myApps directory. For example, if my user was Joe then it might look like this:
/home/joe/openframeworks/apps/myApps/threadgui
Now your ready to start working with ThreadGUI.

3 testApp.cpp/.h

There are two files that, if you just want to use ThreadGUI out of the box, you have to change/write. When you browse through the src file, you'll see theres four files. For those of you that are familiar with Open Frameworks you'll know that main.cpp doesn't need edited unless you want to change the screen settings. testApp.cpp holds an example of all the different functions that are included in ThreadGUI. For example: Program 1

Program 1 Example of the format used

```
TO.roundButtonGreen(400,400,20,"Hello",1);
```

TO is the main ThreadGUI function. If you look at testApp.h you'll see two lines of code that looks like so:

Program 2 testApp.h

```
#include "threadedObject.h"
...
threadedObject TO;
```

This is where you can change the call name if you don't want to use TO.

4 threadedObject.h

This is the main ThreadGUI file, where all of the functions to draw the GUI are.

4.1 buttonAction(int button);

In this function is where you define what all the buttons will do, by button number in a case structure.

4.2 threadedObject();

Here you can set variables to different values.

4.3 start();

Starts the ThreadGUI thread

4.4 stop();

Stops the ThreadGUI thread

4.5 threadedFunction();

4.6 label(int x, int y, char str[20]);

This creates a label at (x,y). the width is determined by the length of str[], and the height is always set at 20px.1

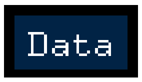

Figure 1: Normal label

4.7 labelGreen(int x, int y, char str[20]);

Same as above but green colored.2

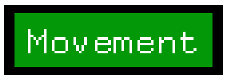

Figure 2: Green label

4.8 labelGrey(int x, int y, char str[20]);

Same as above but grey colored.3

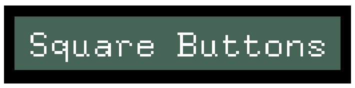

Figure 3: Grey label

4.9 rectButton(int x, int y, char srt[15], int color);

Creates a square button with the center at (x,y), with the text in srt[]. The width and height are automatically calculated from the length of srt[].4

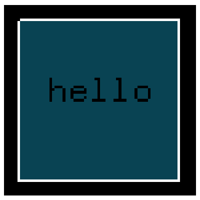

Figure 4: Normal rectButton

4.10 rectButtonGrey(int x, int y, char srt[15], int color);

Same as above but grey colored.5

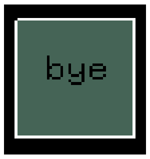

Figure 5: Grey rectButton

4.11 rectButtonGreen(int x, int y, char srt[15], int color);

Same as above but green colored.6

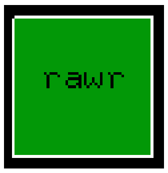

Figure 6: Green rectButton

4.12 roundButton(int x, int y, int h, char srt[15], int color);

Creates a circlular button with the center at (x,y) with radius h and text srt[].7

4.13 roundButtonGrey(int x, int y, int h, char srt[15], int color);

Same as above but grey colored.8

4.14 roundButtonGreen(int x, int y, int h, char srt[15], int color);

Same as above but green colored.9

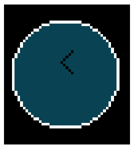

Figure 7: Normal roundButton

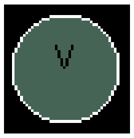

Figure 8: Grey roundButton

4.15 click(int x, int y);

calculates which button has been clicked then passes that button to buttonAction.

4.16 click(int color);

Same as above, but only the button number is passed to it. This could be used for key presses.

4.17 line(int x,int y, int xc, int yc, int thick);

Creates a white line from point (x,y) to (xc,yc) with the thickness thick.10

4.18 lineGreen(int x,int y, int xc, int yc, int thick);

Same as above but green colored.11

4.19 lineGrey(int x,int y, int xc, int yc, int thick);

Same as above but grey colored.12

4.20 lineBlue(int x,int y, int xc, int yc, int thick);

Same as above but blue colored.13

4.21 bullet(int x, int y, char str[20]);

Creates a bulleted list item with the top corner at (x,y) and the text str[].14

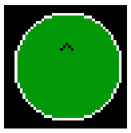
Figure 9: Green roundButton

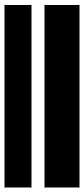
Figure 10: Normal line

Figure 11: Green line

Figure 12: Grey line

Figure 13: Blue line

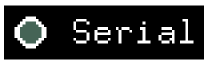

Figure 14: Normal bullet

4.22 bulletGreen(int x, int y, char str[20]);

Same as above but green colored.15

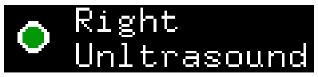

Figure 15: Green bullet

4.23 bulletBlue(int x, int y, char str[20]);

Same as above but blue colored.16

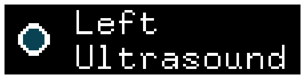

Figure 16: Blue bullet

4.24 mlBullet(int x, int y, char str[20]);

Same as a regular bullet, but this allows for the text to use up 2 lines instead of 1.17

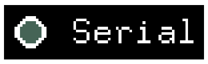

Figure 17: Normal mlBullet

4.25 mlBulletGreen(int x, int y, char str[20]);

Same as above but green colored.18

4.26 mlBulletBlue(int x, int y, char str[20]);

Same as above but blue colored.19

4.27 text(int x, int y, float data);

This ones a little decieving, as it really is a data function, in the ThreadGUI-BOB-GUI branch, this has been renamed to dataFloat. Simply this function takes and places formated text at (x,y+4). The y+4 is so the same coordinates can be used with the bullets and have the text line up still.20

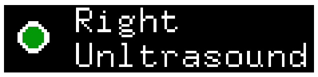

Figure 18: Green mlBullet

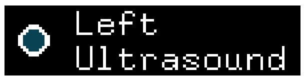

Figure 19: Blue mlBullet

Figure 20: Text