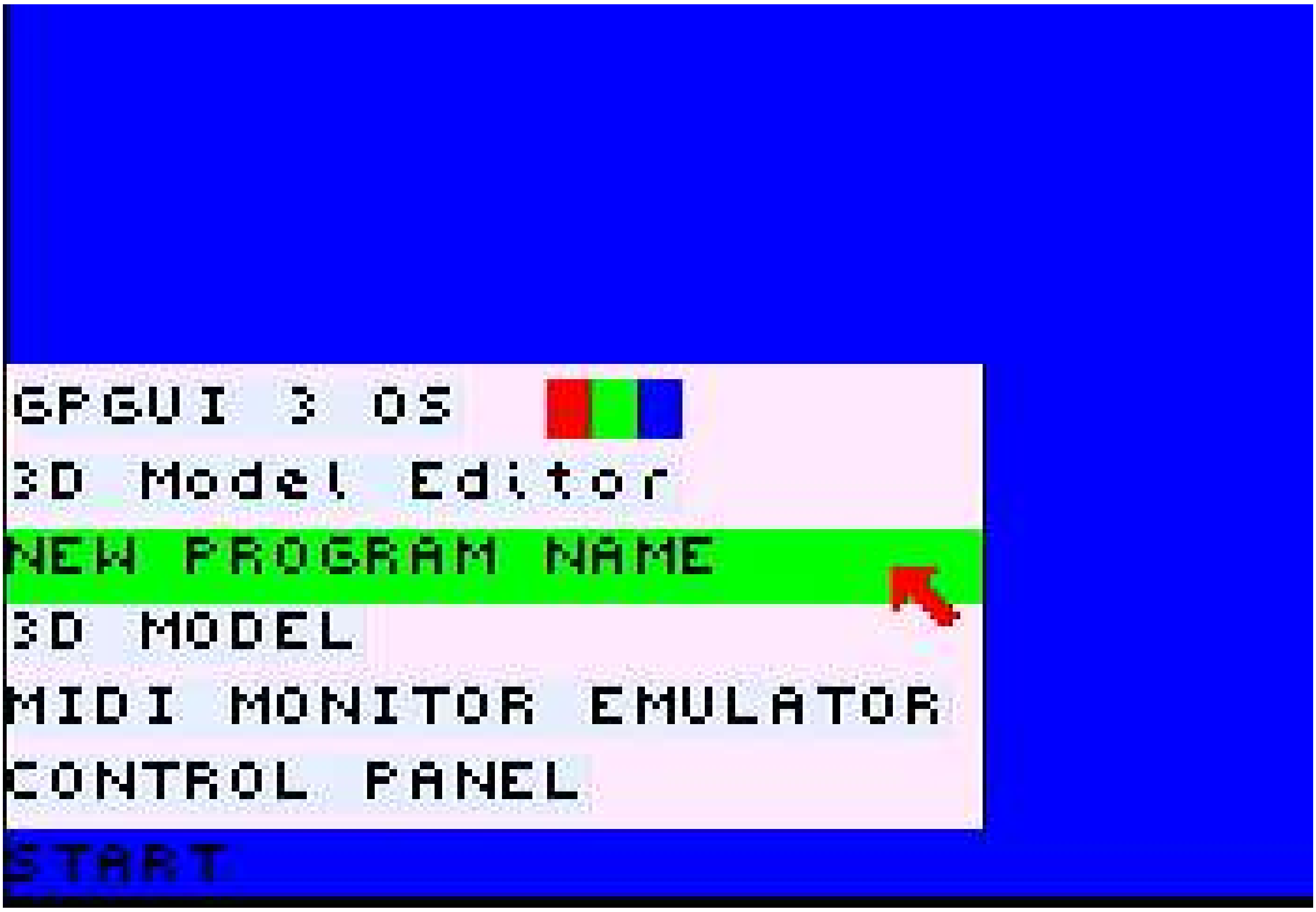
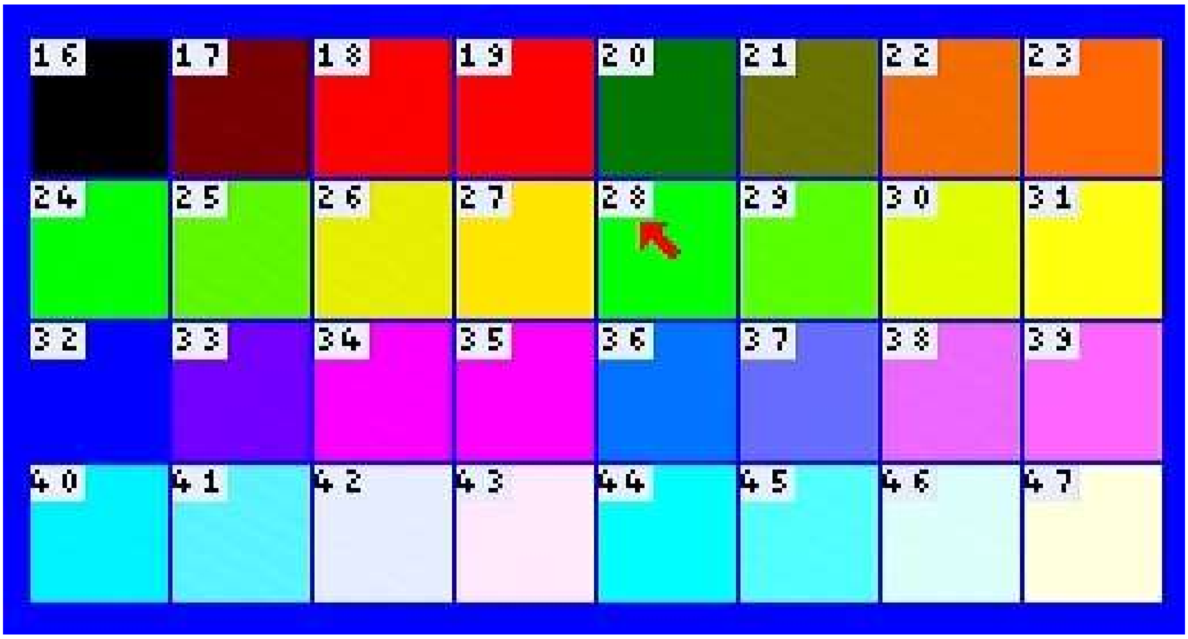
GPGUI 3 - HOW TO CREATE MOUSE BUTTONS

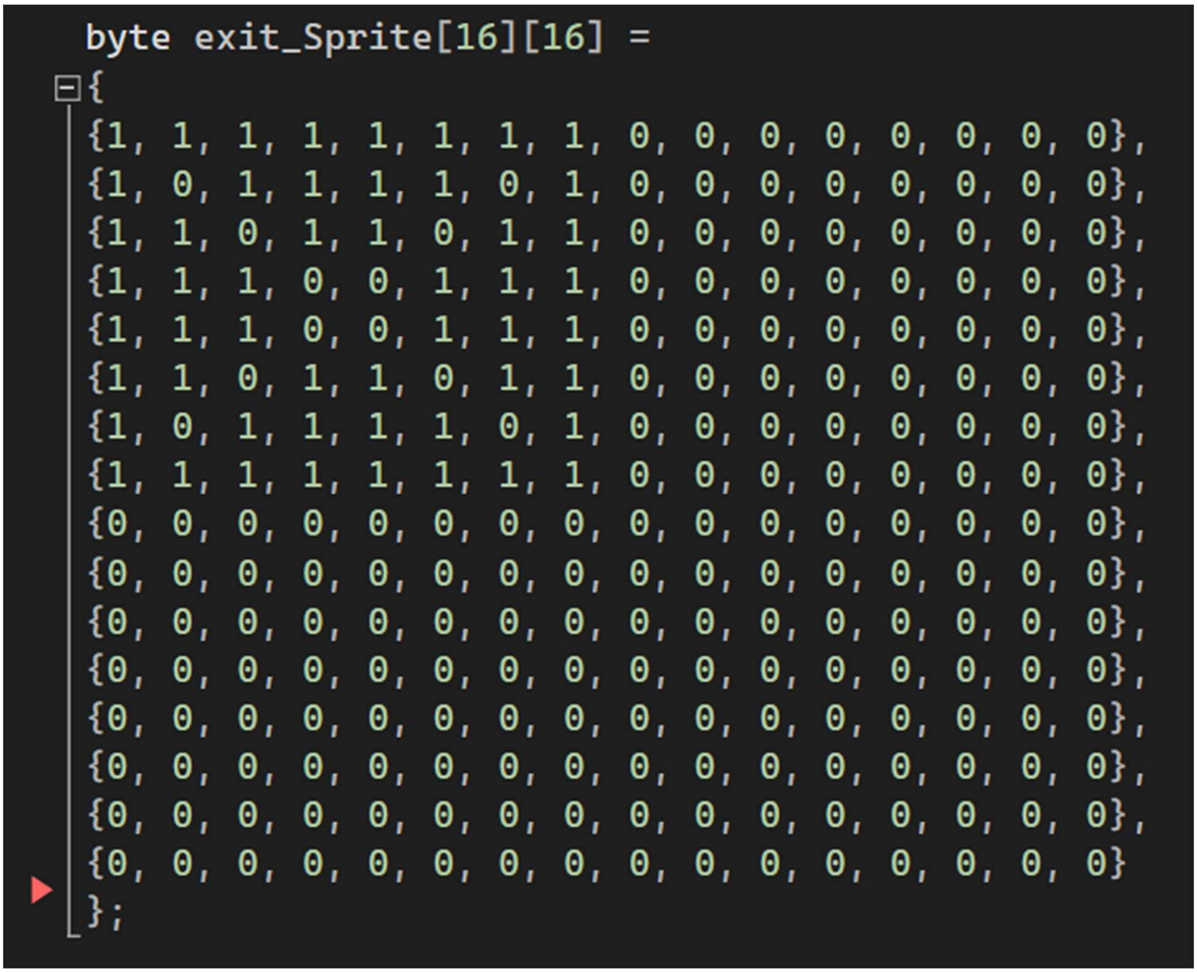


Summary:

How to create basic buttons on the screen that can be clicked on using the mouse cursor. The buttons can be flat shaded ON/OFF and highlight's when the cursor is over it, text that highlight's when the cursor is over it or an icon referenced to a color texture map **byte** array[16][16].

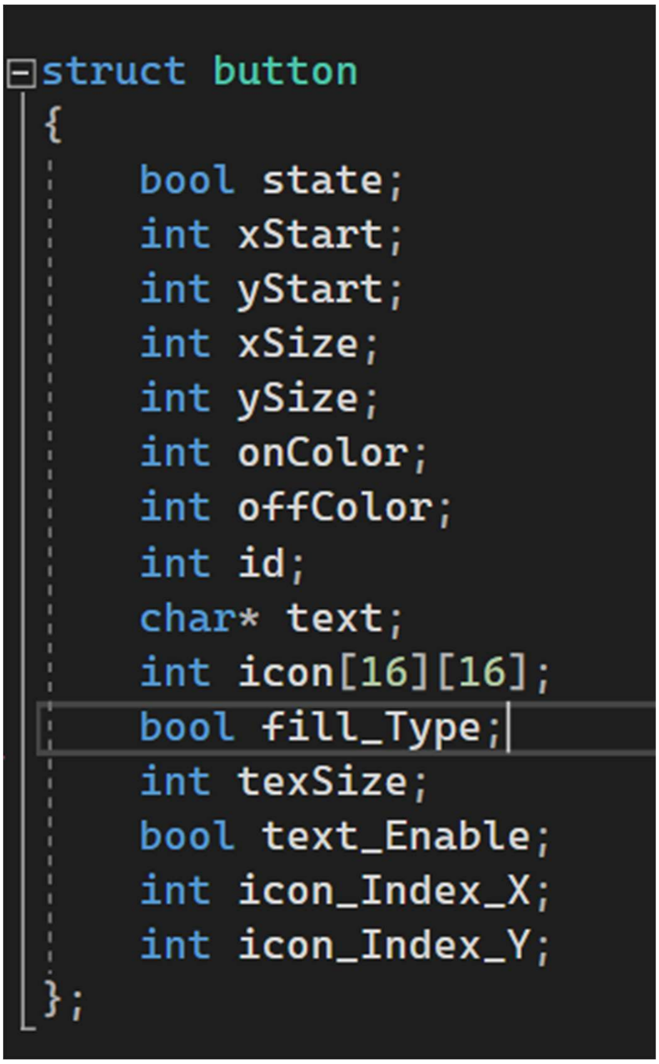
Contents





```
byte exit_Sprite[16][16] =
{
{1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
};
```

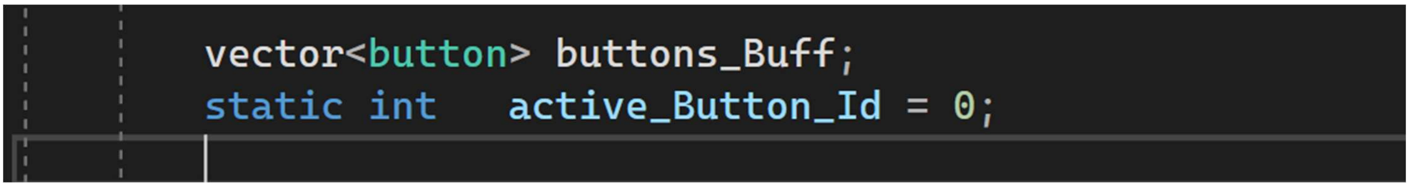
This is the button data structure.



```
struct button
{
    bool state;
    int xStart;
    int yStart;
    int xSize;
    int ySize;
    int onColor;
    int offColor;
    int id;
    char* text;
    int icon[16][16];
    bool fill_Type;
    int texSize;
    bool text_Enable;
    int icon_Index_X;
    int icon_Index_Y;
};
```

STEP 1 - Declare a Vector buffer to hold the buttons.

Create int active_Button_Id memory address and a vector< button > of a struct type **button.**



```
        vector<button> buttons_Buff;
        static int   active_Button_Id = 0;
```

STEP 2 - Create a button object and add it to the buffer (ICON).

Set button start (Top left).

Set button area size.

Set button ID.

Set fill Type to 1.

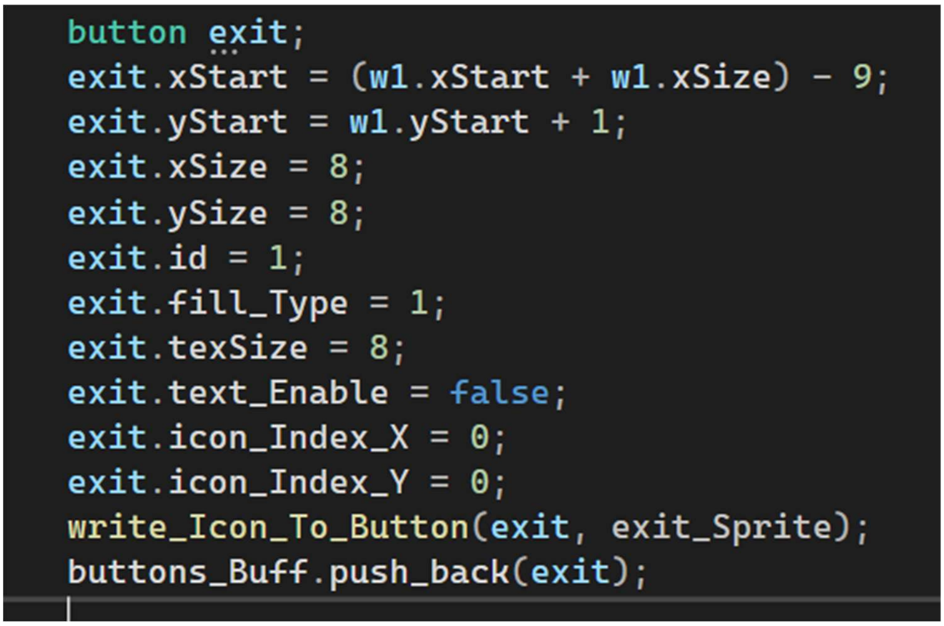
set the texture size (this is a square size) 16x16 max can be used).

Set text Enable too false.

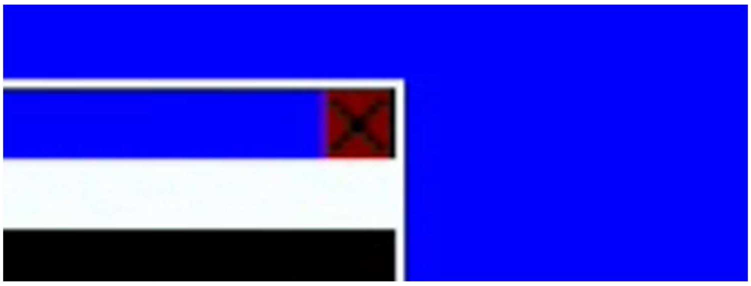
set the texture index (only if the texture is smaller than the button area.

Use the write_Icon_To_Button(**button***, byte*[16][16]) function.

Push the button object into the vector buffer.



```
button exit;
exit.xStart = (w1.xStart + w1.xSize) - 9;
exit.yStart = w1.yStart + 1;
exit.xSize = 8;
exit.ySize = 8;
exit.id = 1;
exit.fill_Type = 1;
exit.texSize = 8;
exit.text_Enable = false;
exit.icon_Index_X = 0;
exit.icon_Index_Y = 0;
write_Icon_To_Button(exit, exit_Sprite);
buttons_Buff.push_back(exit);
```



STEP 2.1 - Create a button object and add to the buffer (FLAT SHADED).

Set the active and inactive color.

Set button start (Top left).

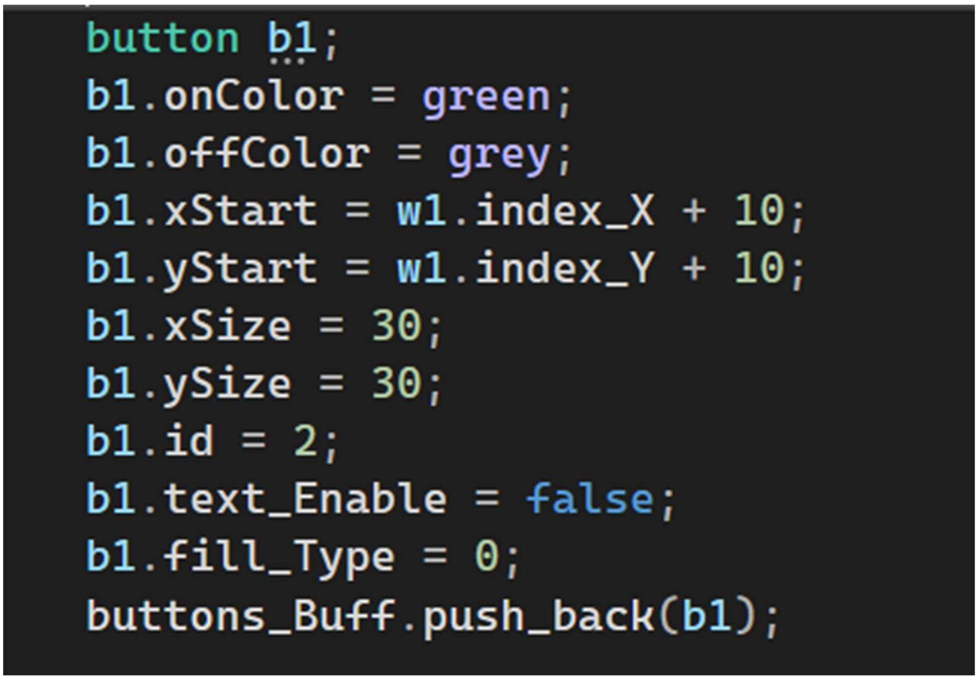
Set the ID.

Set button area size.

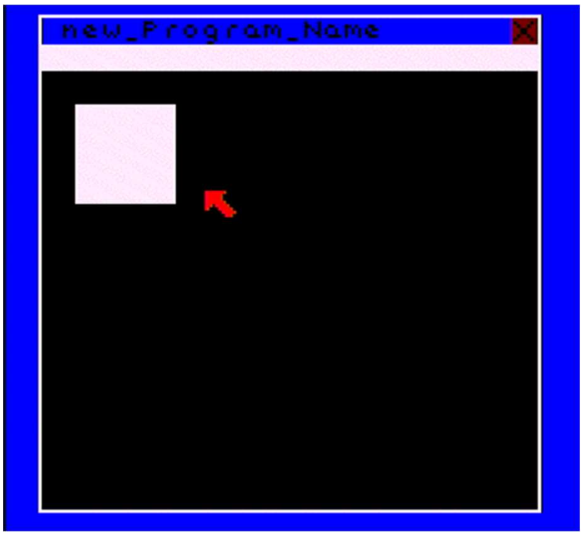
Set text Enable too false.

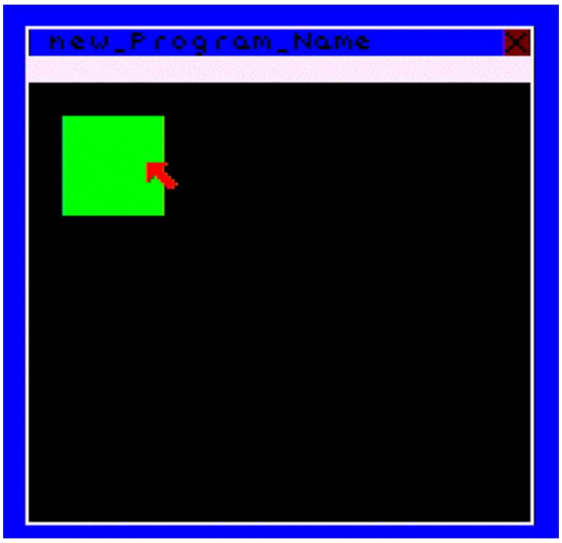
Set fill Type to 0.

Push the button object into the vector buffer.



```
button b1;
b1.onColor = green;
b1.offColor = grey;
b1.xStart = w1.index_X + 10;
b1.yStart = w1.index_Y + 10;
b1.xSize = 30;
b1.ySize = 30;
b1.id = 2;
b1.text_Enable = false;
b1.fill_Type = 0;
buttons_Buff.push_back(b1);
```


new_Program_Name


new_Program_Name