

# Lab2 - FLTK

## Describe assignment:

Use FLTK/FLUID to demonstrate control of a small image (e.g. bug)

**Lab goals:** Explore the use of constructors and a array of object pointers

- Loading the image (read section 9.6)
  - Buttons to allow user to make image move, turn, and/or follow a path.
  - Add default constructor to class to set initial position and direction
  - Declare/define a global object to place in FLTK Box
  - Instead of storing image data directly in executable, load from png file
- Array of objects (read section 9.10)
  - Create 4 images facing in the four directions (N,E,S,W)
  - Create array to hold pointers to these images
  - Copies of images are not allowed (read section 13.5)
- Turning the image
  - In the "turn" button callback, cycle through the 4 images
  - Redraw the parent (window) after each
  - Use local static variable which will remember its previous value
  - sequencing the path of image
- Create button to walk the image through some predetermined path
  - Create array of characters such as "mmt" that means "move, move, turn"
  - Each click of the path button reads next character from string
  - Use local variable to remember last character processed

---

## How to create 4 images facing in the four directions (N,E,S,W)

```
$ convert bugN.png -rotate 90 bugE.png
```

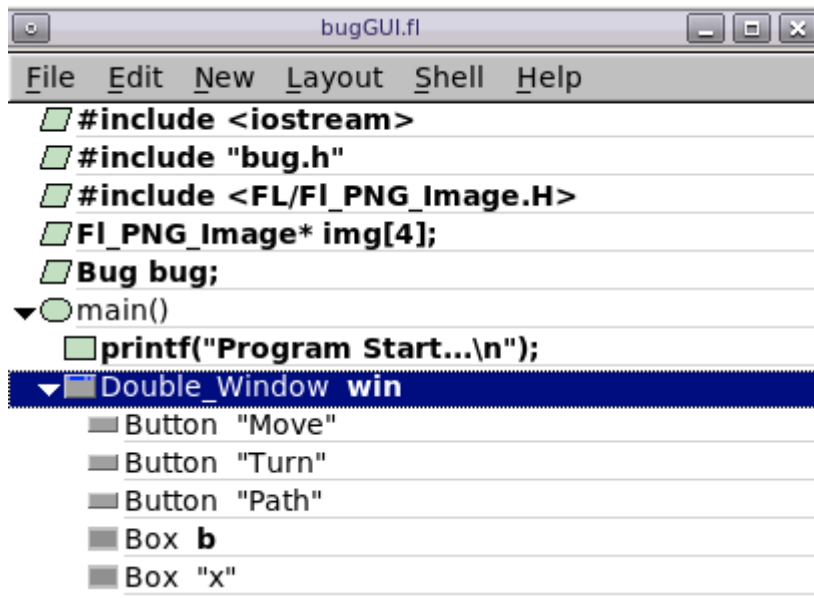
## How to program FLTK with FLUID?

FLTK official documents: [Programming with FLUID](#)

JSLinux: <https://bellard.org/jslinux/vm.html?url=alpine-x86-xwin.cfg&mem=1024&graphic=1&w=1200&h=800>

```
login to sync: `xhost + && vfllogin ACCOUNT`
```

1. Right click mouse in JSLinux, then choose "FLDev"
2. Create bugGUI.fl file by using FLDev interface:



- Create Declaration:
  - #include
  - #include <FL/Fl\_PNG\_Image.h>
  - #include <bug.h>
  - Fl\_PNG\_Image\* img[4]; // Create an an array to hold Fl\_PNG\_Image pointer.
  - Bug bug; // Create an Bug instance as global variable
- Create main() enterpoint:
- Create Double\_Window under/inside main():
  - Create a Box to hold images
  - Create 3 buttoms letting user change position/direction of the images



3. Creating & Setting Box to hold the images:

- Name: b

- Extra Code:

```
![boxGUI_b]
(https://github.com/AmberFu/CS124\_DataStructure\_usingCpp/blob/main/Assignments/pic/bugGlraw=true)
```

#### 4. Creating & Setting each button:

- MOVE: move one unit follow by direction.

**Callback:**

```
bug.move();
b->position(bug.get_row(),bug.get_column())
b->image(img[bug.get_direction()]);
win->redraw();
```

- TURN: turn right for each click.

**Callback:**

```
bug.trun_right();
b->image(img[bug.get_direction()]);
win->redraw();
```

- PATH: follow the path to the end point.

```
bug.path();
b->image(img[bug.get_direction()]);
b->position(bug.get_row(),bug.get_column())
win->redraw();
```

## Create bug.cpp and bug.h:

### bug.h

```
#include <iostream>

using namespace std;

const int NORTH = 0;
const int EAST = 1;
const int SOUTH = 2;
const int WEST = 3;
const int WINDOW_X_MAX = 300;
const int WINDOW_Y_MAX = 200;
const int destinationX = 150;
const int destinationY = 100;

class Bug
```

```

{
public:
    Bug();
    int get_column() const;
    int get_row() const;
    int get_direction() const;
    void trun_right();
    void move();
    void path();
    void go_east(int);
    void go_west(int);
    void go_north(int);
    void go_south(int);
    int destDiff_x();
    int destDiff_y();
private:
    int direction; // 0: North, 1: East, 2: South, 3: West
    int y_position; // 0 - 200
    int x_position; // 0 - 300
};

```

#### bug.cpp

```

#include <cmath>
#include "bug.h"

Bug::Bug()
{
    y_position = 0;
    x_position = 0;
    direction = 1;
}

int Bug::get_column() const
{
    return y_position;
}

int Bug::get_row() const
{
    return x_position;
}

int Bug::get_direction() const
{
    return direction;
}

// TURN
void Bug::trun_right()
{
    if (direction >= 3){

```

```

        direction = 0;
    }else{
        direction++;
    }
}

// MOVE
void Bug::move()
{
    if (direction == NORTH && y_position != 0)
        y_position--;
    else if (direction == EAST && x_position != WINDOW_X_MAX)
        x_position++;
    else if (direction == SOUTH && y_position != WINDOW_Y_MAX)
        y_position++;
    else if (direction == WEST && x_position != 0)
        x_position--;
}

// PATH:
void Bug::go_south(int dest){
    direction = 2; // 0: North, 1: East, 2: South, 3: West
    // if destination is large than 5, move faster ( move 5 unit a time)
    if (std::abs(dest) <= 5){
        y_position += 1;
    }else{
        y_position += 10;
    }
}

void Bug::go_north(int dest){
    direction = 0; // 0: North, 1: East, 2: South, 3: West
    // if destination is large than 5, move faster ( move 5 unit a time)
    if (std::abs(dest) <= 5){
        y_position -= 1;
    }else{
        y_position -= 10;
    }
}

void Bug::go_east(int dest){
    direction = 1; // 0: North, 1: East, 2: South, 3: West
    // if destination is large than 5, move faster ( move 5 unit a time)
    if (std::abs(dest) <= 5){
        x_position += 1;
    }else{
        x_position += 10;
    }
}

void Bug::go_west(int dest){
    direction = 3; // 0: North, 1: East, 2: South, 3: West

```

```

        // if destination is large than 5, move faster ( move 5 unit a time)
        if (std::abs(dest) <= 5 || dest <= -5){
            x_position -= 1;
        }else{
            x_position -= 10;
        }
    }

    int Bug::destDiff_x(){
        return x_position - destinationX;
    }

    int Bug::destDiff_y(){
        return y_position - destinationY;
    }

    void Bug::path()
    {
        int side_of_dest_x = destDiff_x();
        int side_of_dest_y = destDiff_y();
        if (side_of_dest_x < 0){
            go_east(side_of_dest_x);
        }else if (side_of_dest_x > 0){
            go_west(side_of_dest_x);
        }else{
            if (side_of_dest_y > 0){
                go_north(side_of_dest_y);
            }else if (side_of_dest_y < 0){
                go_south(side_of_dest_y);
            }else{
                return;
            }
        }
    }
}

```

- Create .cxx and .h files:

```
$ fluid -c xxx.fl
```

- How to compile bug.cpp with bugGUI.cxx?

```
$ g++ bug.cpp bugGUI.cxx -o bug -lfltk -lXext -lX11 -lm
```

OR use:

```
$ g++ bugGUI.cxx bug.cpp -o bug `fltk-config --cxxflags --ldflags --use-images`
```

- test program:

```
$ ./bug
```

## How to make video in JSLinux?

```
$ ffmpeg -video_size 300x200 -framerate 1 -f x11grab -t 20 -i :0.0 -crf 0 -preset  
ultrafast -c:v libx264rgb bug.mp4
```