Lab 4 Pizza

Generated by Doxygen 1.8.8

Tue Oct 17 2017 14:33:40

Contents

1 Specification

This program will do the following...

2 Analysis

inputs will be:

- The outputs will be:
- The overall algorithim is:

3 order_cb

This is the callback function to order a pizza

Parameters

void	pointers not used

Returns

void

4 File Index

4.1 File List

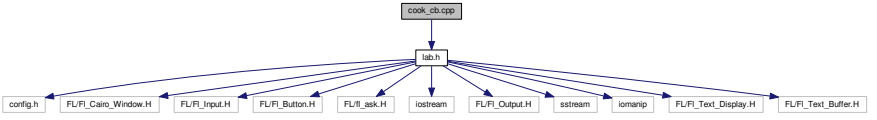
Here is a list of all files with brief descriptions:

5 File Documentation

5.1 cook_cb.cpp File Reference

```
#include "lab.h"
```

Include dependency graph for cook_cb.cpp:



Functions

- void cook_cb (void *)

5.1.1 Function Documentation

5.1.1.1 void cook_cb (void *)

```
3 {
4     //buff->text(pizza->value());
5
6     // temp solution, when it's cooked, put it
7     // in the LLQueue; then here
8     // use your new function that displays
```

```
9     // what is in Q to create a string (with newlines)
10     // in it, return it here and siplay that in the buffer
11
12 }
```

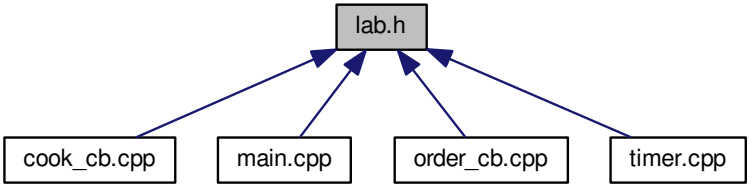
5.2 lab.dox File Reference

5.3 lab.h File Reference

```
#include "config.h"
#include <FL/Fl_Cairo_Window.H>
#include <FL/Fl_Input.H>
#include <FL/Fl_Button.H>
#include <FL/fl_ask.H>
#include <iostream>
#include <FL/Fl_Output.H>
#include <sstream>
#include <iomanip>
#include <FL/Fl_Text_Display.H>
#include <FL/Fl_Text_Buffer.H>
```

Include dependency graph for lab.h:

This graph shows which files directly or indirectly include this file:



Functions

- void order_cb (void *, void *)
- void cook_cb (void *)
- void timer (void *)

Variables

- Fl_Input * pizza
- Fl_Output * watch
- Fl_Text_Buffer * buff
- Fl_Text_Display * orderQ

5.3.1 Function Documentation

5.3.1.1 void cook_cb (void *)

```
3 {
4     //buff->text(pizza->value());
5
6     // temp solution, when it's cooked, put it
7     // in the LLQueue; then here
8     // use your new function that displays
9     // what is in Q to create a string (with newlines)
10     // in it, return it here and siplay that in the buffer
11
12 }
```

5.3.1.2 void order_cb (void *, void *)

```
3 {
4     fl_alert(pizza->value());
5     // cook it
6     Fl::add_timeout(5,cook_cb);
7 }
```

5.3.1.3 void timer (void *)

```
3 {
4     //std::cout << "1 sec" << std::endl;
5     static int s= 0; static int m = 0;
6     std::ostringstream oss; // don't discard the memory.
7                                                 // keep it so we can update t to the window
8     s++;    if(s == 59) {s = 0; m++; }
```

```
9      oss << std::setfill('0');
10      oss << std::setw(2) << m << ":" << std::setw(2) << s;
11      //oss << s;
12      watch->value(oss.str().c_str());
13
14      // Here we could check if the Q's have pizza and drivers
15      // ready for delivery every 10 seconds so we can see order
16      // in Q      if (s % 10 == 0)
17      static std::string str;
18      std::string pizzas[] = {"veggie","pepperoni","sausage","hawaiian"};
19      if(s % 10 == 0)
20      {
21          str += pizzas[s%4] += "\n";
22          buff->text(str.c_str());
23      }
24
25      Fl::repeat_timeout(1,timer);
26  }
```

5.3.2 Variable Documentation

5.3.2.1 Fl_Text_Buffer* buff

5.3.2.2 Fl_Text_Display* orderQ

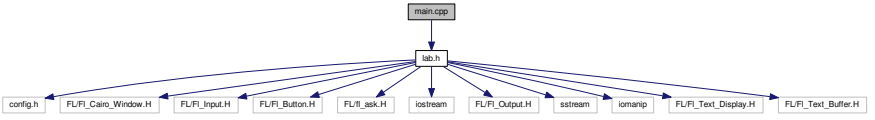
5.3.2.3 Fl_Input* pizza

5.3.2.4 Fl_Output* watch

5.4 main.cpp File Reference

```
#include "lab.h"
```

Include dependency graph for main.cpp:



Functions

- int main ()

Variables

- Fl_Input * pizza
- Fl_Output * watch
- Fl_Text_Buffer * buff
- Fl_Text_Display * orderQ

5.4.1 Function Documentation

5.4.1.1 int main ( )

```
10 {
```

```
11      Fl_Cairo_Window cw(400,300); // width & height of window
12      cw.label("Pizza Deliveries Extravaganja"); // title of your cairo window
13      //cw.color(FL_GREEN);
14
15      pizza = new Fl_Input(190, 20, 100, 20, "pizza:");
16      pizza->labelcolor(FL_BLUE);
17
18      buff = new Fl_Text_Buffer();
19      orderQ = new Fl_Text_Display(100,100,100,100,"Order Q");
20      orderQ->buffer(buff);
21
22      watch = new Fl_Output(70,20,50,20,"seconds:");
23
24      Fl_Button b(330, 60, 50, 20, "Order:");
25      b.callback((Fl_Callback*)order_cb);
26
27      cw.show();
28      Fl::add_timeout(1,timer);
29      return Fl::run();
30  }
```

5.4.2 Variable Documentation

5.4.2.1 Fl_Text_Buffer* buff

5.4.2.2 Fl_Text_Display* orderQ

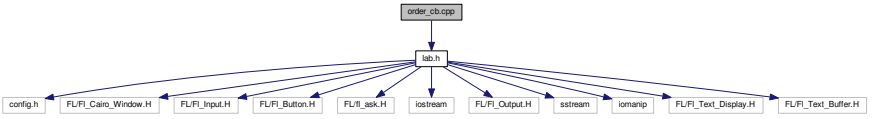
5.4.2.3 Fl_Input* pizza

5.4.2.4 Fl_Output* watch

5.5 order_cb.cpp File Reference

```
#include "lab.h"
```

Include dependency graph for order_cb.cpp:



Functions

- void order_cb (void *, void *)

5.5.1 Function Documentation

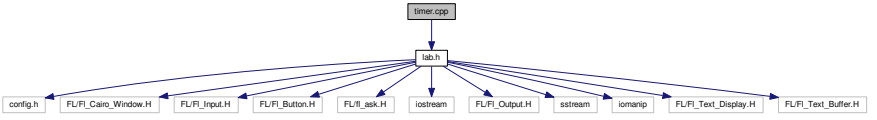
5.5.1.1 void order_cb (void *, void *)

```
3 {
4     fl_alert(pizza->value());
5     // cook it
6     Fl::add_timeout(5,cook_cb);
7 }
```

5.6 timer.cpp File Reference

```
#include "lab.h"
```

Include dependency graph for timer.cpp:



Functions

- void timer (void *)

5.6.1 Function Documentation

5.6.1.1 void timer (void *)

```
3 {
4     //std::cout << "1 sec" << std::endl;
5     static int s= 0; static int m = 0;
6     std::ostringstream oss; // don't discard the memory.
7                                                 // keep it so we can update t to the window
8     s++;     if(s == 59) {s = 0; m++; }
9     oss << std::setfill('0');
10     oss << std::setw(2) << m << ":" << std::setw(2) << s;
11     //oss << s;
```

```
12      watch->value(oss.str().c_str());
13
14      // Here we could check if the Q's have pizza and drivers
15      // ready for delivery every 10 seconds so we can see order
16      // in Q     if (s % 10 == 0)
17      static std::string str;
18      std::string pizzas[] = {"veggie","pepperoni","sausage","hawaiian"};
19      if(s % 10 == 0)
20      {
21          str += pizzas[s%4] += "\n";
22          buff->text(str.c_str());
23      }
24
25      Fl::repeat_timeout(1,timer);
26  }
```