Reference Manual

Generated by Doxygen 1.8.13

Contents

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

2 File Index

2.1 File List

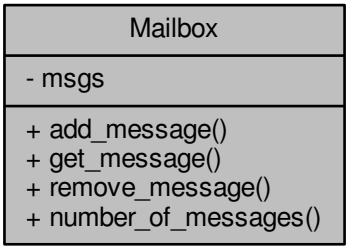
Here is a list of all files with brief descriptions:

3 Class Documentation

3.1 Mailbox Class Reference

```
#include <Mailbox.h>
```

Collaboration diagram for Mailbox:



Public Member Functions

- void add_message (Message m)
- Message get_message (int i) const
- void remove_message (int i)
- int number_of_messages () const

Private Attributes

- std::vector< Message > msgs

3.1.1 Detailed Description

Mailbox class is built for holding the multiple messages. functions are to add and remove messages.

Definition at line 14 of file Mailbox.h.

3.1.2 Member Function Documentation

3.1.2.1 add_message()

```
void Mailbox::add_message (
            Message m )
```

Parameters

m	the Message that is send to the Mailbox

Definition at line 9 of file Mailbox.cxx.

```
9                                           {
10    msgs.push_back(m);
11  }
```

Here is the caller graph for this function:

3.1.2.2 get_message()

```
Message Mailbox::get_message (
            int i ) const
```

Parameters

i	the position of Mailbox to print Message

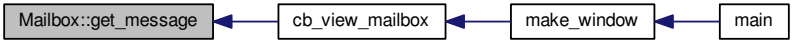
Returns

returns the Message at the specific position

Definition at line 19 of file Mailbox.cxx.

```
19                                                  {
20    return msgs[i];
21  }
```

Here is the caller graph for this function:



3.1.2.3 number_of_messages()

```
int Mailbox::number_of_messages ( ) const
```

Returns

returns the total number of messages in vector

Definition at line 36 of file Mailbox.cxx.

```
36                                                  {
37    return msgs.size();
38 }
```

Here is the caller graph for this function:

3.1.2.4 remove_message()

```
void Mailbox::remove_message (
            int i )
```

removes message from vector

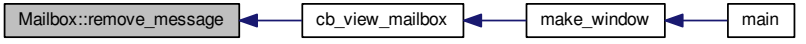
Parameters

i	the position of Mailbox to remove Message

Definition at line 28 of file Mailbox.cxx.

```
28                                          {
29   msgs.erase(msgs.begin()+i*sizeof(Message));
30 }
```

Here is the caller graph for this function:



3.1.3 Member Data Documentation

3.1.3.1 msgs

```
std::vector<Message> Mailbox::msgs  [private]
```

Definition at line 21 of file Mailbox.h.

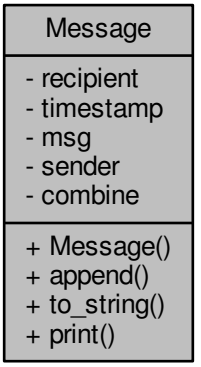
The documentation for this class was generated from the following files:

- Mailbox.h
- Mailbox.cxx

3.2 Message Class Reference

```
#include <Message.h>
```

Collaboration diagram for Message:



Public Member Functions

- Message (std::string s, std::string r)
- void append (std::string s)
- void to_string ()
- bool print (int i)

Private Attributes

- std::string recipient
- time_t timestamp
- std::string msg
- std::string sender
- std::string combine

3.2.1 Detailed Description

ctime is used to output current time used to access fl_choice function This is the Message Class which holds one message

Definition at line 20 of file Message.h.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Message()

```
Message::Message (
            std::string s,
            std::string r )
```

Parameters

s	the name of person sending message
r	the name of person recieving the message

Definition at line 10 of file Message.cxx.

```
10                                                  {
11   sender = s;
12   recipient = r;
13   timestamp = time(0);
14 }
```

3.2.3 Member Function Documentation

3.2.3.1 append()

```
void Message::append (
            std::string s )
```

Parameters

s	the string that you are appending to the message

Definition at line 20 of file Message.cxx.

```
20                                          {
21    msg += s + "\n";
22 }
```

Here is the caller graph for this function:

3.2.3.2 print()

```
bool Message::print (
            int i )
```

the print function aims to fl_choice an alert for user to view the message. if it was called from the Mailbox, it has options of Keep and Remove if it was called from lab view_message button, it print with Ok.

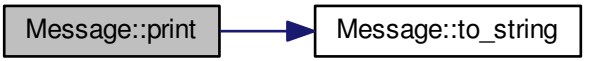
Parameters

i	the position of the vector of messages and is -1 if it is not from Mailbox

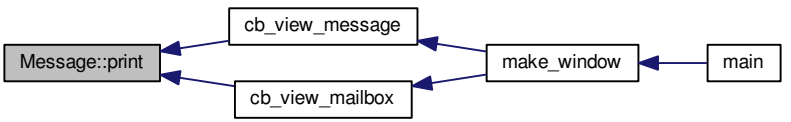
Definition at line 48 of file Message.cxx.

```
48                              {
49   to_string();
50   int keep;
51   if (i == -1)
52     fl_choice(combine.c_str(),"Ok",0,0);
53   else
54     keep = fl_choice(combine.c_str(),"Remove","Keep",0);
55
56   if (i != -1 && keep == 0)
57     return false;
58   return true;
59 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



3.2.3.3 to_string()

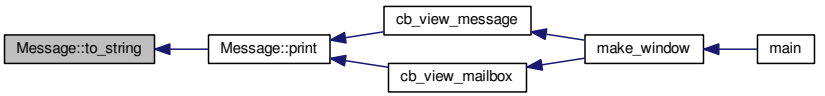
```
void Message::to_string ( )
```

This function converts the msg(message) into a string with a name of combine. Adds sender, recipient, and time to combine

Definition at line 29 of file Message.cxx.

```
29                          {
30   combine = "From: " + sender + "\n";
31   combine += "To: " + recipient + "\n";
32   std::string time = ctime(&timestamp);
33   combine += "Time sent: " + time;
34
35   combine += msg;
36 }
```

Here is the caller graph for this function:



3.2.4 Member Data Documentation

3.2.4.1 combine

```
std::string Message::combine  [private]
```

Definition at line 36 of file Message.h.

3.2.4.2 msg

```
std::string Message::msg  [private]
```

Definition at line 34 of file Message.h.

3.2.4.3 recipient

```
std::string Message::recipient  [private]
```

Definition at line 27 of file Message.h.

3.2.4.4 sender

```
std::string Message::sender  [private]
```

Definition at line 35 of file Message.h.

3.2.4.5 timestamp

```
time_t Message::timestamp  [private]
```

timestamp is of type time_t to use function ctime to output char array of current time

Definition at line 33 of file Message.h.

The documentation for this class was generated from the following files:

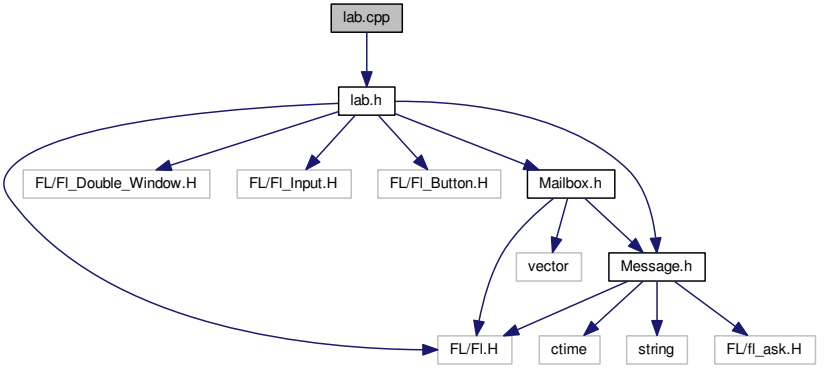
- Message.h
- Message.cxx

4 File Documentation

4.1 lab.cpp File Reference

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

Include dependency graph for lab.cpp:



Functions

- int main ()

4.1.1 Function Documentation

4.1.1.1 main()

```
int main ( )
```

Figure 1 Compose_before

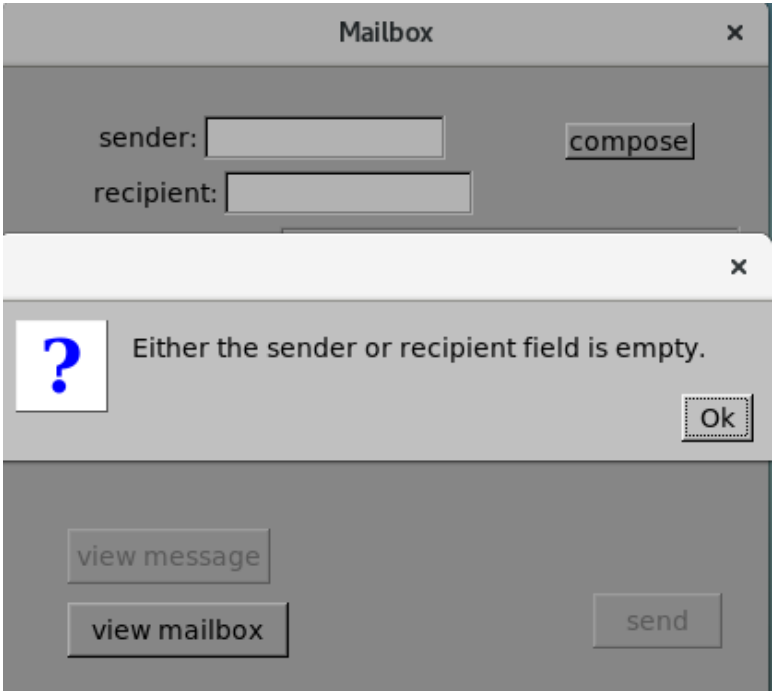


Figure 2 Compose_after

Figure 3 Message1

Figure 4 Message3

Figure 5 Message4

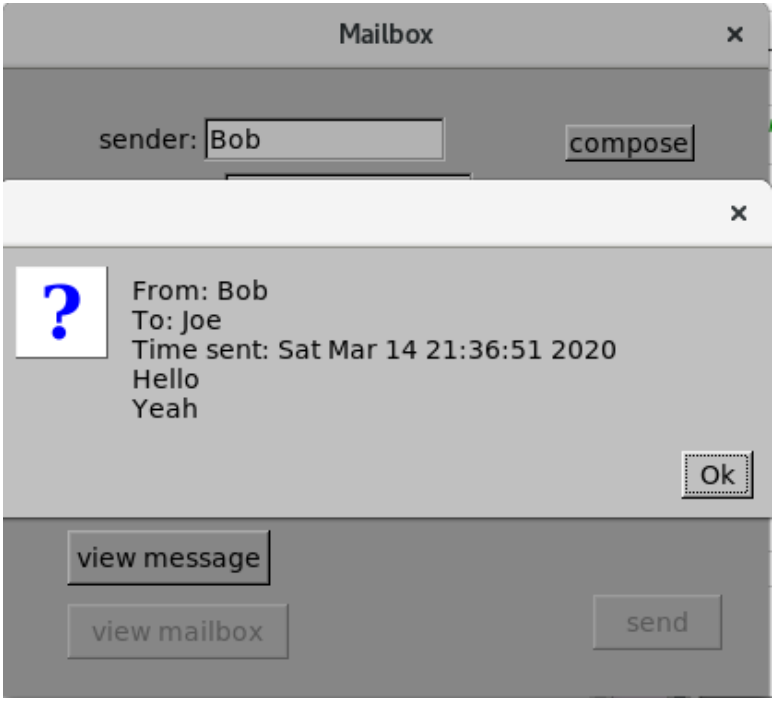


Figure 6 view_message

Figure 7 Send_before

Figure 8 Send_after

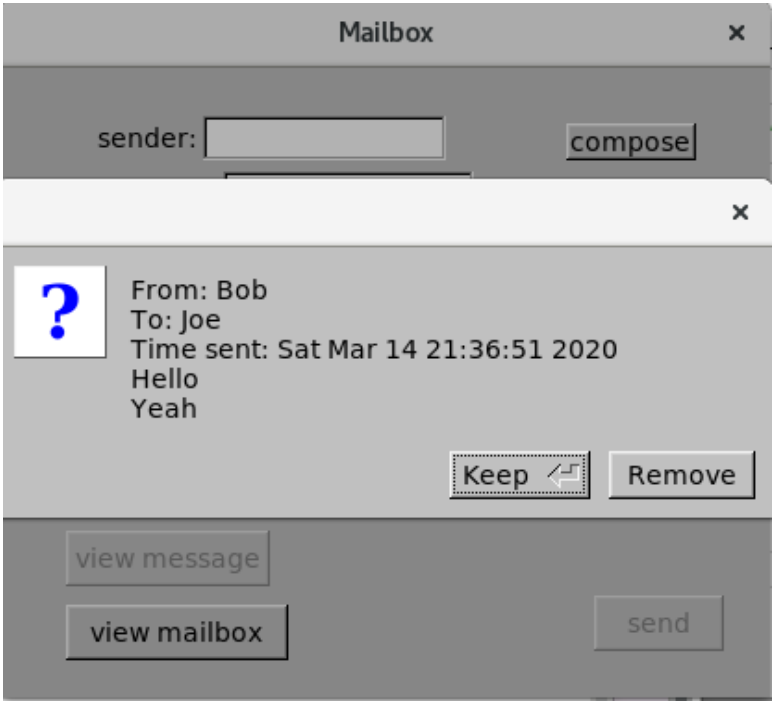


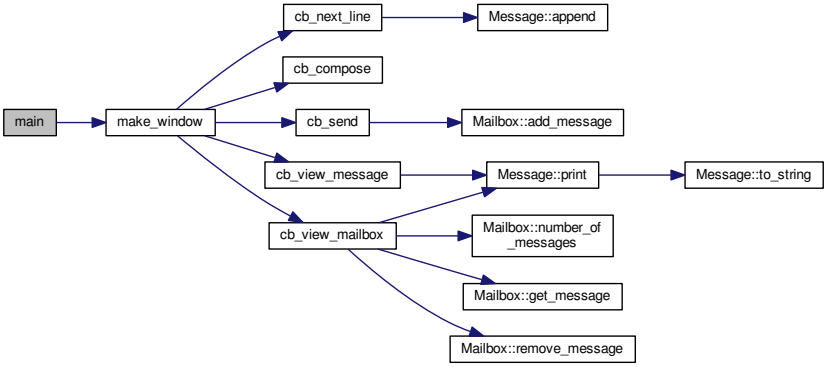
Figure 9 view_mailbox

Figure 10 view_mailbox2

Definition at line 14 of file lab.cpp.

```
15 {
16     make_window()->show();
17     Fl::run();
18 }
```

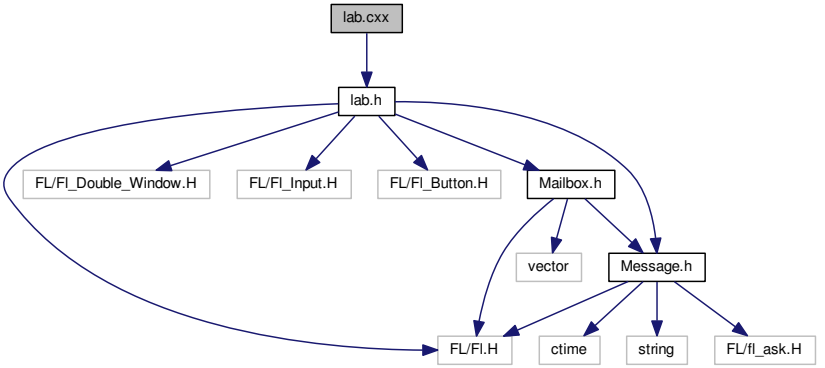
Here is the call graph for this function:



4.2 lab.cxx File Reference

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

Include dependency graph for lab.cxx:



Functions

- static void cb_next_line (Fl_Button *, void *)
- static void cb_compose (Fl_Button *, void *)
- static void cb_send (Fl_Button *, void *)
- static void cb_view_message (Fl_Button *, void *)
- static void cb_view_mailbox (Fl_Button *, void *)
- Fl_Double_Window * make_window ()

Variables

- static std::string missing_send_rec = "Either the sender or recipient field is empty."
- static std::string missing_message = "The message field is empty."
- static std::string no_mail = "There is no mail."
- Fl_Double_Window * w =(Fl_Double_Window *)0
- Fl_Input * sender =(Fl_Input *)0
- Fl_Input * recipient =(Fl_Input *)0
- Fl_Input * message =(Fl_Input *)0
- Fl_Button * next_line =(Fl_Button *)0
- Fl_Button * send =(Fl_Button *)0
- Fl_Button * view_message =(Fl_Button *)0
- Fl_Button * view_mailbox =(Fl_Button *)0
- Message * mp
- Mailbox mb

4.2.1 Function Documentation

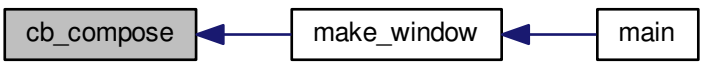
4.2.1.1 cb_compose()

```
static void cb_compose (
            Fl_Button * ,
            void *  )  [static]
```

Definition at line 46 of file lab.cxx.

```
46                                                  {
47   std::string test = sender->value();
48 std::string test2 = recipient->value();
49 if (test != "" && test2 != "")
50 {
51     mp = new Message(sender->value(),recipient->value());
52     message->activate();
53     next_line->activate();
54 }
55 else
56     fl_choice(missing_send_rec.c_str(),"Ok",0,0);
57 }
```

Here is the caller graph for this function:



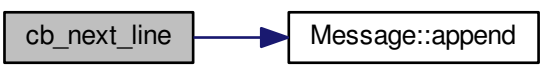
4.2.1.2 cb_next_line()

```
static void cb_next_line (
            Fl_Button * ,
            void *  )  [static]
```

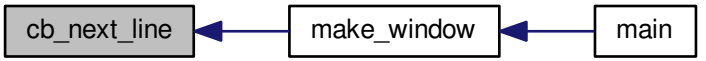
Definition at line 34 of file lab.cxx.

```
34                                                  {
35   std::string test = message->value();
36 if (test != "")
37 {
38 mp->append(message->value());
39 message->value("");
40 view_message->activate();
41 }
42 else
43     fl_choice(missing_message.c_str(),"Ok",0,0);
44 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.1.3 cb_send()

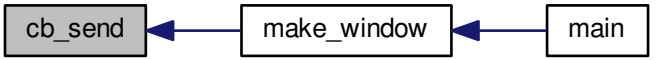
```
static void cb_send (
            Fl_Button * ,
            void *  )  [static]
```

Definition at line 61 of file lab.cxx.

```
61                                                  {
62   mb.add_message(*mp);
63
64 sender->value("");
65 recipient->value("");
66
67 view_mailbox->activate();
68 message->deactivate();
69 next_line->deactivate();
70 view_message->deactivate();
71 send->deactivate();
72 }
```

Here is the call graph for this function:

Here is the caller graph for this function:



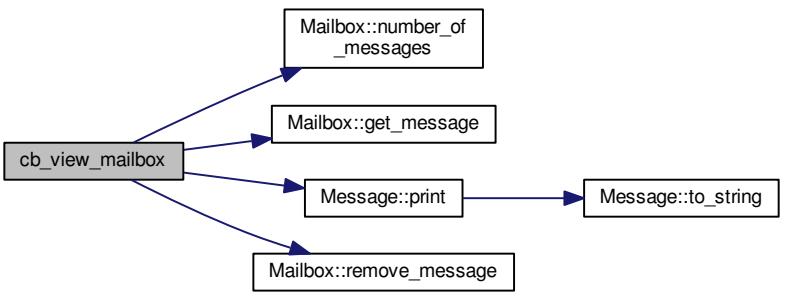
4.2.1.4 cb_view_mailbox()

```
static void cb_view_mailbox (
            Fl_Button * ,
            void *  )  [static]
```

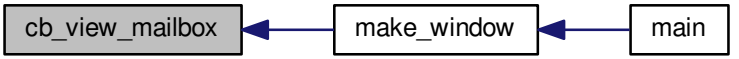
Definition at line 84 of file lab.cxx.

```
84                                                          {
85   if (!mb.number_of_messages())
86     fl_choice(no_mail.c_str(),"Ok",0,0);
87
88
89 for(int i = 0;i < mb.number_of_messages();i++)
90 {
91     Message temp = mb.get_message(i);
92     bool keep = temp.print(i);
93     if (!keep)
94     {
95         mb.remove_message(i);
96     }
97 };
98 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



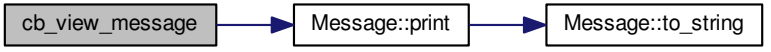
4.2.1.5 cb_view_message()

```
static void cb_view_message (
            Fl_Button * ,
            void *  )  [static]
```

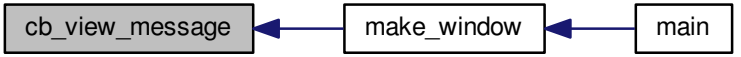
Definition at line 76 of file lab.cxx.

```
76                                                          {
77   bool useless = mp->print(-1);
78
79 send->activate();
80 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.1.6 make_window()

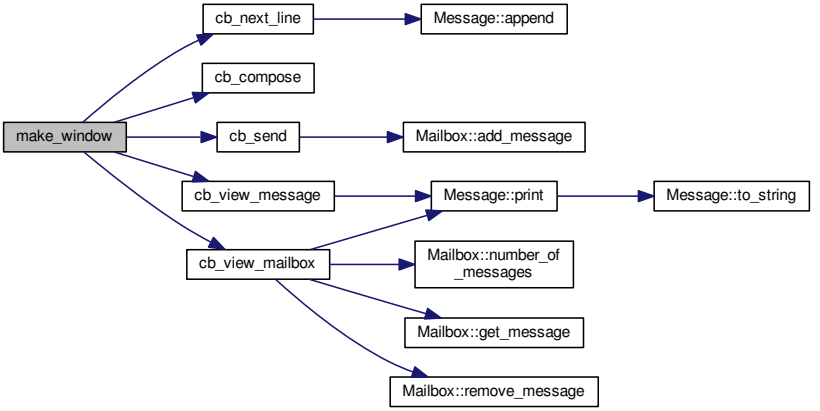
```
Fl_Double_Window* make_window ( )
```

Definition at line 100 of file lab.cxx.

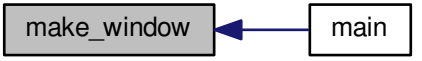
```
100                                     {
101   { w = new Fl_Double_Window(415, 340, "Mailbox");
```

```
102     { sender = new Fl_Input(109, 26, 131, 24, "sender:");
103     } // Fl_Input* sender
104     { recipient = new Fl_Input(120, 56, 135, 24, "recipient:");
105     } // Fl_Input* recipient
106     { message = new Fl_Input(150, 86, 250, 23, "message line:");
107       message->deactivate();
108     } // Fl_Input* message
109     { next_line = new Fl_Button(50, 120, 70, 20, "next line");
110       next_line->tooltip("Click here to enter new line of message");
111       next_line->callback((Fl_Callback*)cb_next_line);
112       next_line->deactivate();
113     } // Fl_Button* next_line
114     { Fl_Button* o = new Fl_Button(305, 30, 70, 20, "compose");
115       o->callback((Fl_Callback*)cb_compose);
116     } // Fl_Button* o
117     { send = new Fl_Button(320, 285, 70, 30, "send");
118       send->callback((Fl_Callback*)cb_send);
119       send->deactivate();
120     } // Fl_Button* send
121     { view_message = new Fl_Button(35, 250, 110, 30, "view message");
122       view_message->callback((Fl_Callback*)cb_view_message);
123       view_message->deactivate();
124     } // Fl_Button* view_message
125     { view_mailbox = new Fl_Button(35, 290, 120, 30, "view mailbox");
126       view_mailbox->callback((Fl_Callback*)cb_view_mailbox);
127       view_mailbox->deactivate();
128     } // Fl_Button* view_mailbox
129     w->end();
130   } // Fl_Double_Window* w
131   return w;
132 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2 Variable Documentation

4.2.2.1 mb

```
Mailbox mb
```

This Mailbox object is global using default constructor Will aggragate messages into a vector

Definition at line 144 of file lab.cxx.

4.2.2.2 message

```
Fl_Input* message =(Fl_Input *)0
```

Definition at line 30 of file lab.cxx.

4.2.2.3 missing_message

```
std::string missing_message = "The message field is empty."  [static]
```

this string is error message for missing field of message. "The message field is empty."

Definition at line 16 of file lab.cxx.

4.2.2.4 missing_send_rec

```
std::string missing_send_rec = "Either the sender or recipient field is empty."  [static]
```

this string is error message for missing fields of sender or recipient. "Either the sender or recipient field is empty."

Definition at line 10 of file lab.cxx.

4.2.2.5 mp

```
Message* mp
```

This Message object is global since it will be accesed from more than one source code file.

Definition at line 138 of file lab.cxx.

4.2.2.6 next_line

```
Fl_Button* next_line =(Fl_Button *)0
```

Definition at line 32 of file lab.cxx.

4.2.2.7 no_mail

```
std::string no_mail = "There is no mail."  [static]
```

this string is error message for when there are no mail in Mailbox. "The message field is empty."

Definition at line 22 of file lab.cxx.

4.2.2.8 recipient

```
Fl_Input* recipient =(Fl_Input *)0
```

Definition at line 28 of file lab.cxx.

4.2.2.9 send

```
Fl_Button* send =(Fl_Button *)0
```

Definition at line 59 of file lab.cxx.

4.2.2.10 sender

```
Fl_Input* sender =(Fl_Input *)0
```

Definition at line 26 of file lab.cxx.

4.2.2.11 view_mailbox

```
Fl_Button* view_mailbox =(Fl_Button *)0
```

Definition at line 82 of file lab.cxx.

4.2.2.12 view_message

```
Fl_Button* view_message =(Fl_Button *)0
```

Definition at line 74 of file lab.cxx.

4.2.2.13 w

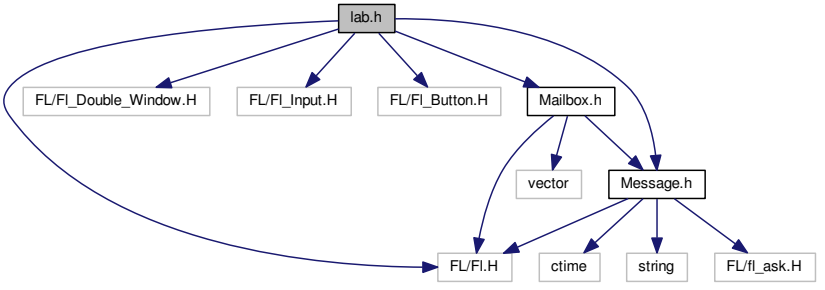
```
Fl_Double_Window* w =(Fl_Double_Window *)0
```

Definition at line 24 of file lab.cxx.

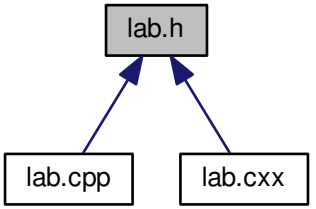
4.3 lab.h File Reference

```
#include <FL/Fl.H>
#include <FL/Fl_Double_Window.H>
#include <FL/Fl_Input.H>
#include <FL/Fl_Button.H>
#include "Message.h"
#include "Mailbox.h"
```

Include dependency graph for lab.h:



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



Functions

- Fl_Double_Window * make_window ()

Variables

- Fl_Double_Window * w
- Fl_Input * sender
- Fl_Input * recipient
- Fl_Input * message
- Fl_Button * next_line
- Fl_Button * send
- Fl_Button * view_message
- Fl_Button * view_mailbox
- Message * mp
- Mailbox mb

4.3.1 Function Documentation

4.3.1.1 make_window()

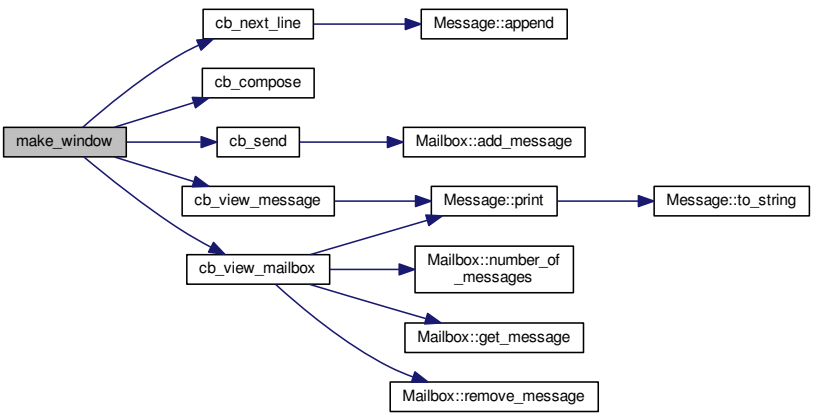
```
Fl_Double_Window* make_window ( )
```

Definition at line 100 of file lab.cxx.

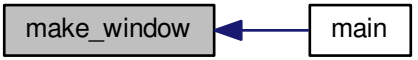
```
100                                     {
101   { w = new Fl_Double_Window(415, 340, "Mailbox");
102     { sender = new Fl_Input(109, 26, 131, 24, "sender:");
103     } // Fl_Input* sender
104     { recipient = new Fl_Input(120, 56, 135, 24, "recipient:");
105     } // Fl_Input* recipient
106     { message = new Fl_Input(150, 86, 250, 23, "message line:");
107       message->deactivate();
108     } // Fl_Input* message
109     { next_line = new Fl_Button(50, 120, 70, 20, "next line");
110       next_line->tooltip("Click here to enter new line of message");
111       next_line->callback((Fl_Callback*)cb_next_line);
112       next_line->deactivate();
113     } // Fl_Button* next_line
114     { Fl_Button* o = new Fl_Button(305, 30, 70, 20, "compose");
115       o->callback((Fl_Callback*)cb_compose);
116     } // Fl_Button* o
117     { send = new Fl_Button(320, 285, 70, 30, "send");
118       send->callback((Fl_Callback*)cb_send);
119       send->deactivate();
120     } // Fl_Button* send
```

```
121     { view_message = new Fl_Button(35, 250, 110, 30, "view message");
122       view_message->callback((Fl_Callback*)cb_view_message);
123       view_message->deactivate();
124     } // Fl_Button* view_message
125     { view_mailbox = new Fl_Button(35, 290, 120, 30, "view mailbox");
126       view_mailbox->callback((Fl_Callback*)cb_view_mailbox);
127       view_mailbox->deactivate();
128     } // Fl_Button* view_mailbox
129     w->end();
130   } // Fl_Double_Window* w
131   return w;
132 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.2 Variable Documentation

4.3.2.1 mb

```
Mailbox mb
```

This Mailbox object is global using default constructor Will aggragate messages into a vector

Definition at line 144 of file lab.cxx.

4.3.2.2 message

```
Fl_Input* message
```

Definition at line 30 of file lab.cxx.

4.3.2.3 mp

```
Message* mp
```

This Message object is global since it will be accesed from more than one source code file.

Definition at line 138 of file lab.cxx.

4.3.2.4 next_line

```
Fl_Button* next_line
```

Definition at line 32 of file lab.cxx.

4.3.2.5 recipient

```
Fl_Input* recipient
```

Definition at line 28 of file lab.cxx.

4.3.2.6 send

```
Fl_Button* send
```

Definition at line 59 of file lab.cxx.

4.3.2.7 sender

```
Fl_Input* sender
```

Definition at line 26 of file lab.cxx.

4.3.2.8 view_mailbox

```
Fl_Button* view_mailbox
```

Definition at line 82 of file lab.cxx.

4.3.2.9 view_message

```
Fl_Button* view_message
```

Definition at line 74 of file lab.cxx.

4.3.2.10 w

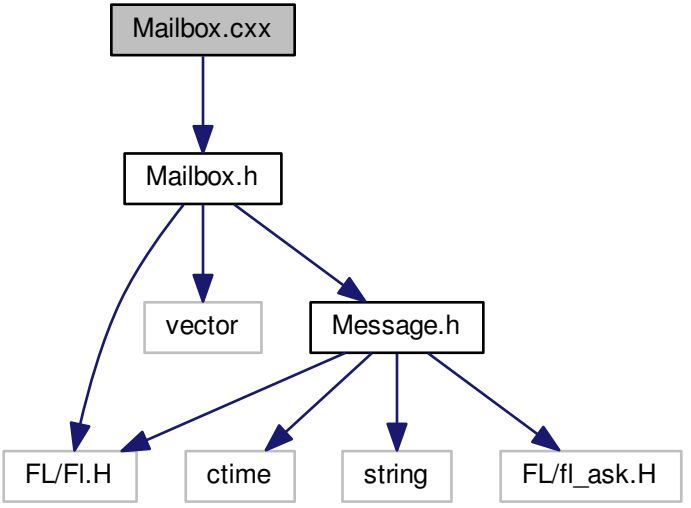
```
Fl_Double_Window* w
```

Definition at line 24 of file lab.cxx.

4.4 Mailbox.cxx File Reference

```
#include "Mailbox.h"
```

Include dependency graph for Mailbox.cxx:

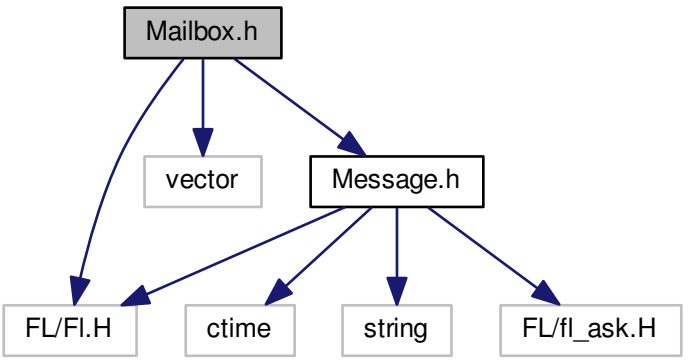


4.5 Mailbox.h File Reference

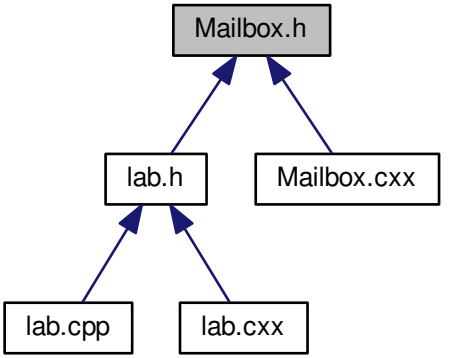
```
#include <FL/Fl.H>
#include <vector>
```

```
#include "Message.h"
```

Include dependency graph for Mailbox.h:



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



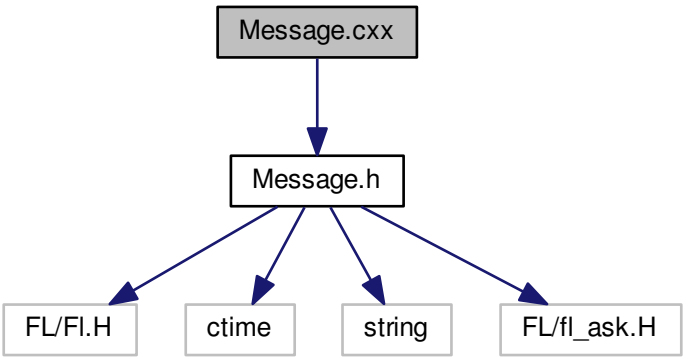
Classes

- class Mailbox

4.6 Message.cxx File Reference

```
#include "Message.h"
```

Include dependency graph for Message.cxx:

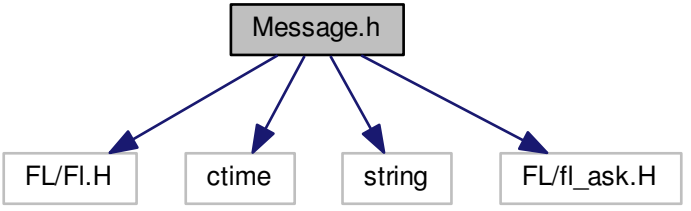


4.7 Message.h File Reference

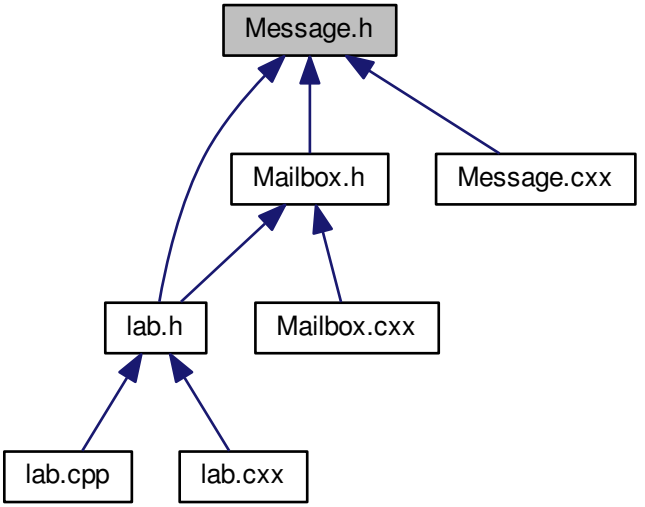
```
#include <FL/Fl.H>
#include <ctime>
```

```
#include <string>
#include <FL/fl_ask.H>
```

Include dependency graph for Message.h:



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



Classes

- class Message

Index