Bachelorthesis - Multi-/Remotemaus

Generated by Doxygen 1.8.13

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 input_event Struct Reference

```
#include <osinput.h>
```

Public Attributes

- timeval time
- uint16_t type
- uint16_t code
- int32_t value

3.1.1 Detailed Description

dummy structer for OSX

3.1.2 Member Data Documentation

3.1.2.1 code

```
uint16_t input_event::code
```

3.1.2.2 time

```
timeval input_event::time
```

3.1.2.3 type

```
uint16_t input_event::type
```

3.1.2.4 value

```
int32_t input_event::value
```

The documentation for this struct was generated from the following file:

- inc/osinput.h

3.2 mrmCfgParser Class Reference

```
#include <mrmCfgParser.h>
```

Public Member Functions

- mrmCfgParser (string jsonstring)
- json parse ()
- list< mrmMouse > createLocalMice (int ∗count)
- list< mrmCursor > createLocalCursors (int ∗count)
- list< mrmCursor > createRemoteCursors (int ∗count)
- list< mrmScreen > createScreens (int ∗count)
- int getUDPPort ()
- string getHostIP ()

Public Attributes

- string jsonstr
- json jsonobj

3.2.1 Constructor & Destructor Documentation

3.2.1.1 mrmCfgParser()

```
mrmCfgParser::mrmCfgParser (
             string jsonstring )
```

creates a cfg parser. Add the text to be parsed as a parameter

Parameters

jsonstring	

3.2.2 Member Function Documentation

3.2.2.1 createLocalCursors()

```
list< mrmCursor > mrmCfgParser::createLocalCursors (
            int * count )
```

creates a list of all the configured local cursors

Parameters

count	

Returns

list of the cursor objects

3.2.2.2 createLocalMice()

```
list< mrmMouse > mrmCfgParser::createLocalMice (
            int * count )
```

creates a list of all the configured local mice

Parameters

count	

Returns

list of the mice objects

3.2.2.3 createRemoteCursors()

```
list< mrmCursor > mrmCfgParser::createRemoteCursors (
            int * count )
```

creates a list of all the configured remote cursors

Parameters

count	

Returns

list of the cursor objects

3.2.2.4 createScreens()

```
list< mrmScreen > mrmCfgParser::createScreens (
            int * count )
```

creates a list of all the configured remote screens

Parameters

count	

Returns

list of the screen objects

3.2.2.5 getHostIP()

```
string mrmCfgParser::getHostIP ( )
```

return the configured host ip as a string

Returns

3.2.2.6 getUDPPort()

```
int mrmCfgParser::getUDPPort ( )
```

return the configured UDP port

Returns

3.2.2.7 parse()

```
json mrmCfgParser::parse ( )
```

parse the received string to an json object

Returns

3.2.3 Member Data Documentation

3.2.3.1 jsonobj

```
json mrmCfgParser::jsonobj
```

3.2.3.2 jsonstr

```
string mrmCfgParser::jsonstr
```

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

- inc/mrmCfgParser.h
- src/mrmCfgParser.cpp

3.3 mrmCursor Class Reference

```
#include <mrmCursor.h>
```

Public Member Functions

- int getPosx () const
- void setPosx (int posx)
- int getPosy () const
- void setPosy (int posy)
- int getXMax ()
- int getYMax ()
- Display ∗ getDisplay ()
- mrmCursor (string id, int color_r, int color_g, int color_b, int show, int click, int datatf, bool active, int devicenumbers[ ])
- string getid ()
- void run (mrmCursor ∗cursor)
- void start ()
- int updateTheMessageQueue ()
- void createTheWindow ()
- void createTheRenderContext ()
- void redrawTheWindow ()
- void describe_fbconfig (GLXFBConfig fbconfig)
- void setlastabsx (int lastabs)
- void setlastabsy (int lastabs)
- int getrelxfromabs ()
- int getrelyfromabs ()
- bool isactive ()
- void activate (bool activate)

Public Attributes

- int clickrunning
- string id
- int color_r
- int color_g
- int color_b
- int show
- int click
- int datatf
- int lastabsx
- int lastabsy
- int relx
- int rely

Private Attributes

- int Xscreen
- Atom del_atom
- Colormap cmap
- Display ∗ Xdisplay
- XVisualInfo ∗ visual
- XRenderPictFormat ∗ pict_format
- GLXFBConfig ∗ fbconfigs
- GLXFBConfig fbconfig
- int numfbconfigs
- GLXContext render_context
- Window Xroot
- Window window_handle
- GLXWindow glX_window_handle
- int width
- int height
- int posx
- int posy
- bool active

Static Private Attributes

- static int VisData [ ]

3.3.1 Constructor & Destructor Documentation

3.3.1.1 mrmCursor()

```
mrmCursor::mrmCursor (
             string id,
             int color_r,
             int color_g,
             int color_b,
             int show,
             int click,
             int datatf,
             bool active,
             int devicenumbers[] )
```

create the current cursor with al the needed configuration values

Parameters

id	
color_r	
color_g	
color_b	
show	
click	
datatf	
active	
devicenumbers	

3.3.2 Member Function Documentation

3.3.2.1 activate()

```
void mrmCursor::activate (
             bool activate )
```

activate or deactivate the cursor. The parameter defines if deactivated or activated

Parameters

activate	

3.3.2.2 createTheRenderContext()

```
void mrmCursor::createTheRenderContext ( )
```

creating the needed render content to be able to draw the window and the cursor

3.3.2.3 createTheWindow()

```
void mrmCursor::createTheWindow ( )
```

create the window needed to draw in it

3.3.2.4 describe_fbconfig()

```
void mrmCursor::describe_fbconfig (
             GLXFBConfig fbconfig )
```

set/describe the current used fbconfig

Returns

3.3.2.5 getDisplay()

```
Display * mrmCursor::getDisplay ( )
```

3.3.2.6 getid()

```
string mrmCursor::getid ( )
```

get the id of this cursor as string

Returns

3.3.2.7 getPosx()

```
int mrmCursor::getPosx ( ) const
```

get position

Returns

3.3.2.8 getPosy()

```
int mrmCursor::getPosy ( ) const
```

get position

Returns

3.3.2.9 getrelxfromabs()

```
int mrmCursor::getrelxfromabs ( )
```

calculate the relative value from the last two absolut values

Returns

rel value

3.3.2.10 getrelyfromabs()

```
int mrmCursor::getrelyfromabs ( )
```

calculate the relative value from the last two absolut values

Returns

rel value

3.3.2.11 getXMax()

```
int mrmCursor::getXMax ( )
```

get max position

Returns

3.3.2.12 getYMax()

```
int mrmCursor::getYMax ( )
```

get max position

Returns

3.3.2.13 isactive()

```
bool mrmCursor::isactive ( )
```

returns the current activ state

Returns

3.3.2.14 redrawTheWindow()

```
void mrmCursor::redrawTheWindow ( )
```

redraw the window which holds our cursor. Redraw the cursor into this window.

3.3.2.15 run()

```
void mrmCursor::run (
             mrmCursor * cursor )
```

3.3.2.16 setlastabsx()

```
void mrmCursor::setlastabsx (
             int lastabs )
```

set the last absolut value

Parameters

lastabs	

3.3.2.17 setlastabsy()

```
void mrmCursor::setlastabsy (
             int lastabs )
```

set the last absolut value

Parameters

lastabs	

3.3.2.18 setPosx()

```
void mrmCursor::setPosx (
             int posx )
```

set position

Returns

3.3.2.19 setPosy()

```
void mrmCursor::setPosy (
             int posy )
```

set position

Returns

3.3.2.20 start()

```
void mrmCursor::start ( )
```

starts the Thread which is running the cursor. The run function wil be called in the thread.

3.3.2.21 updateTheMessageQueue()

```
int mrmCursor::updateTheMessageQueue ( )
```

Check the MEssage queue of the windowserver. Act if needed

Returns

3.3.3 Member Data Documentation

3.3.3.1 active

```
bool mrmCursor::active  [private]
```

3.3.3.2 click

```
int mrmCursor::click
```

3.3.3.3 clickrunning

```
int mrmCursor::clickrunning
```

3.3.3.4 cmap

```
Colormap mrmCursor::cmap  [private]
```

3.3.3.5 color_b

```
int mrmCursor::color_b
```

3.3.3.6 color_g

```
int mrmCursor::color_g
```

3.3.3.7 color_r

```
int mrmCursor::color_r
```

3.3.3.8 datatf

```
int mrmCursor::datatf
```

3.3.3.9 del_atom

```
Atom mrmCursor::del_atom  [private]
```

3.3.3.10 fbconfig

```
GLXFBConfig mrmCursor::fbconfig  [private]
```

3.3.3.11 fbconfigs

```
GLXFBConfig* mrmCursor::fbconfigs  [private]
```

3.3.3.12 glX_window_handle

```
GLXWindow mrmCursor::glX_window_handle  [private]
```

3.3.3.13 height

```
int mrmCursor::height  [private]
```

3.3.3.14 id

```
string mrmCursor::id
```

3.3.3.15 lastabsx

```
int mrmCursor::lastabsx
```

3.3.3.16 lastabsy

```
int mrmCursor::lastabsy
```

3.3.3.17 numfbconfigs

```
int mrmCursor::numfbconfigs  [private]
```

3.3.3.18 pict_format

```
XRenderPictFormat* mrmCursor::pict_format  [private]
```

3.3.3.19 posx

```
int mrmCursor::posx  [private]
```

3.3.3.20 posy

```
int mrmCursor::posy  [private]
```

3.3.3.21 relx

```
int mrmCursor::relx
```

3.3.3.22 rely

```
int mrmCursor::rely
```

3.3.3.23 render_context

```
GLXContext mrmCursor::render_context  [private]
```

3.3.3.24 show

```
int mrmCursor::show
```

3.3.3.25 VisData

```
int mrmCursor::VisData  [static], [private]
```

Initial value:

```
= {
        GLX_RENDER_TYPE, GLX_RGBA_BIT,
        GLX_DRAWABLE_TYPE, GLX_WINDOW_BIT,
        GLX_DOUBLEBUFFER, True,
        GLX_RED_SIZE, 8,
        GLX_GREEN_SIZE, 8,
        GLX_BLUE_SIZE, 8,
        GLX_ALPHA_SIZE, 8,
        GLX_DEPTH_SIZE, 16,
        None
}
```

3.3.3.26 visual

```
XVisualInfo* mrmCursor::visual  [private]
```

3.3.3.27 width

```
int mrmCursor::width  [private]
```

3.3.3.28 window_handle

```
Window mrmCursor::window_handle  [private]
```

3.3.3.29 Xdisplay

```
Display* mrmCursor::Xdisplay  [private]
```

3.3.3.30 Xroot

```
Window mrmCursor::Xroot  [private]
```

3.3.3.31 Xscreen

```
int mrmCursor::Xscreen  [private]
```

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

- inc/mrmCursor.h
- src/mrmCursor.cpp

3.4 mrmCursorControl Class Reference

```
#include <mrmCursorControl.h>
```

Public Member Functions

- mrmCursorControl (list< mrmCursor > localcursors, list< mrmCursor > remotecursors, int amountlc, int amountrc)
- void start (mrmOSCursor ∗)
- list< mrmCursor > getlist ()
- int work_event (json work_event)

Public Attributes

- list< mrmCursor > cursors
- int amount
- mrmOSCursor ∗ oscursor
- unsigned long amountlocal

3.4.1 Constructor & Destructor Documentation

3.4.1.1 mrmCursorControl()

```
mrmCursorControl::mrmCursorControl (
             list< mrmCursor > localcursors,
             list< mrmCursor > remotecursors,
             int amountlc,
             int amountrc )
```

3.4.2 Member Function Documentation

3.4.2.1 getlist()

```
list<mrmCursor> mrmCursorControl::getlist ( )
```

3.4.2.2 start()

```
void mrmCursorControl::start (
             mrmOSCursor ∗ oscursor )
```

starts all the Thread of the known mice

3.4.2.3 work_event()

```
int mrmCursorControl::work_event (
             json work_event )
```

receives an event to work with. The event will be processed and given to the correct object

Parameters

work_event	

Returns

3.4.3 Member Data Documentation

3.4.3.1 amount

```
int mrmCursorControl::amount
```

3.4.3.2 amountlocal

```
unsigned long mrmCursorControl::amountlocal
```

3.4.3.3 cursors

```
list<mrmCursor> mrmCursorControl::cursors
```

3.4.3.4 oscursor

```
mrmOSCursor* mrmCursorControl::oscursor
```

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

- inc/mrmCursorControl.h
- src/mrmCursorControl.cpp

3.5 mrmMiceControl Class Reference

```
#include <mrmMiceControl.h>
```

Public Member Functions

- mrmMiceControl (list< mrmMouse > localmice, int amount)
- void start ()

Public Attributes

- list< mrmMouse > mice
- int amount

3.5.1 Constructor & Destructor Documentation

3.5.1.1 mrmMiceControl()

```
mrmMiceControl::mrmMiceControl (
             list< mrmMouse > localmice,
             int amount )
```

3.5.2 Member Function Documentation

3.5.2.1 start()

```
void mrmMiceControl::start ( )
```

starts all the Thread of the known mice

3.5.3 Member Data Documentation

3.5.3.1 amount

```
int mrmMiceControl::amount
```

3.5.3.2 mice

```
list<mrmMouse> mrmMiceControl::mice
```

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

- inc/mrmMiceControl.h
- src/mrmMiceControl.cpp

3.6 mrmMouse Class Reference

```
#include <mrmMouse.h>
```

Public Member Functions

- mrmMouse (string id, string evenfile)
- void start ()
- void run (mrmMouse ∗mouse)

Public Attributes

- struct input_event ∗ ieMouse
- int mousefile
- const char ∗ id
- const char ∗ eventfile

3.6.1 Constructor & Destructor Documentation

3.6.1.1 mrmMouse()

```
mrmMouse::mrmMouse (
            string id,
            string evenfile )
```

create the mouse object. It will know which Devicefile to read with the eventfile string parameter.

Parameters

id	
evenfile	

3.6.2 Member Function Documentation

3.6.2.1 run()

```
void mrmMouse::run (
             mrmMouse * mouse )
```

the run function. This function will bi called from the mouse thread. The run function receives Mousevent packages and send them to our controller

Parameters

mouse	

3.6.2.2 start()

```
void mrmMouse::start ( )
```

starts the Thread which is running the mousobject. The run function wil be called in the thread.

3.6.3 Member Data Documentation

3.6.3.1 eventfile

```
const char* mrmMouse::eventfile
```

3.6.3.2 id

```
const char* mrmMouse::id
```

3.6.3.3 ieMouse

```
struct input_event* mrmMouse::ieMouse
```

3.6.3.4 mousefile

```
int mrmMouse::mousefile
```

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

- inc/mrmMouse.h
- src/mrmMouse.cpp

3.7 mrmMultiRemoteMouse Class Reference

```
#include <mrmMultiRemoteMouse.h>
```

Public Member Functions

- mrmMultiRemoteMouse (list< mrmMouse > localmice, int amountlm, list< mrmCursor > localcursors, int amountlc, list< mrmCursor > remotecursors, int amountrc, list< mrmScreen > remotescreens, int amountrs, mrmUDPClient ∗udpclient, mrmUDPServer ∗udpserver)
- void start ()
- void recvMouseEvent (json mouseEvent)
- void sendscreenchange (json scevent)
- void sendmouseevent (json mevent)
- string getscreen (int i)
- string gethostip ()

Public Attributes

- list< mrmScreen > screens
- mrmMiceControl ∗ micectrl
- mrmCursorControl ∗ cursorctrl
- mrmOSCursor ∗ oscursor
- int amountscreen
- int tempcount
- string hostip

Private Attributes

- mrmUDPClient ∗ client
- mrmUDPServer ∗ server

3.7.1 Constructor & Destructor Documentation

3.7.1.1 mrmMultiRemoteMouse()

```
mrmMultiRemoteMouse::mrmMultiRemoteMouse (
             list< mrmMouse > localmice,
             int amountlm,
             list< mrmCursor > localcursors,
             int amountlc,
             list< mrmCursor > remotecursors,
             int amountrc,
             list< mrmScreen > remotescreens,
             int amountrs,
             mrmUDPClient * udpclient,
             mrmUDPServer * udpserver )
```

creates the multimouse controller. The multimouse controller needs to know all the objects. All known objects are given with the constructor

Parameters

localmice	
amountlm	
localcursors	
amountlc	
remotecursors	
amountrc	
remotescreens	
amountrs	
udpclient	
udpserver	

3.7.2 Member Function Documentation

3.7.2.1 gethostip()

```
string mrmMultiRemoteMouse::gethostip ( )
```

this function returns the host IP as string value

Returns

3.7.2.2 getscreen()

```
string mrmMultiRemoteMouse::getscreen (
             int i )
```

Sometimes the cursor needs to get the next screen in a specific direction. The cursor will call this function. The controller knows the screen objects and will answer with an IP as a string

Parameters

i	

Returns

screen IP as string value

3.7.2.3 recvMouseEvent()

```
void mrmMultiRemoteMouse::recvMouseEvent (
            json mouseEvent )
```

If a mouse or a server receives a mouseevent, it will call this function. This function will send the event to the correct controler and work the event.

Parameters

mouseEvent	

3.7.2.4 sendmouseevent()

```
void mrmMultiRemoteMouse::sendmouseevent (
            json mevent )
```

if a cursor needs to send a mouseevent it will call this function. The controller knows the client and will forward the request. The request is given with the parameter json object

Parameters

mevent	

3.7.2.5 sendscreenchange()

```
void mrmMultiRemoteMouse::sendscreenchange (
            json scevent )
```

if a cursor needs to send a screenchange it will call this function. The controller knows the client and will forward the request. The request is given with the parameter json object

Parameters

scevent	

3.7.2.6 start()

```
void mrmMultiRemoteMouse::start ( )
```

starting all threads of the integreated objects

3.7.3 Member Data Documentation

3.7.3.1 amountscreen

```
int mrmMultiRemoteMouse::amountscreen
```

3.7.3.2 client

```
mrmUDPClient* mrmMultiRemoteMouse::client  [private]
```

3.7.3.3 cursorctrl

```
mrmCursorControl* mrmMultiRemoteMouse::cursorctrl
```

3.7.3.4 hostip

```
string mrmMultiRemoteMouse::hostip
```

3.7.3.5 micectrl

```
mrmMiceControl* mrmMultiRemoteMouse::micectrl
```

3.7.3.6 oscursor

```
mrmOSCursor* mrmMultiRemoteMouse::oscursor
```

3.7.3.7 screens

```
list<mrmScreen> mrmMultiRemoteMouse::screens
```

3.7.3.8 server

```
mrmUDPServer* mrmMultiRemoteMouse::server  [private]
```

3.7.3.9 tempcount

```
int mrmMultiRemoteMouse::tempcount
```

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

- inc/mrmMultiRemoteMouse.h
- src/mrmMultiRemoteMouse.cpp

3.8 mrmOSCursor Class Reference

```
#include <mrmOSCursor.h>
```

Public Member Functions

- mrmOSCursor ()
- void getMutex ()
- void releaseMutex ()
- void setCursor (int posx, int posy)
- void clickPress (int key)
- void clickRelease (int key)

Public Attributes

- int posx
- int posy
- Display ∗ dpy
- pthread_mutex_t mutex

3.8.1 Constructor & Destructor Documentation

3.8.1.1 mrmOSCursor()

```
mrmOSCursor::mrmOSCursor ( )
```

create the OS cursor object

3.8.2 Member Function Documentation

3.8.2.1 clickPress()

```
void mrmOSCursor::clickPress (
             int key )
```

simulates a button click press on the os cursor. The param gives the used key for the click

Parameters

key	

3.8.2.2 clickRelease()

```
void mrmOSCursor::clickRelease (
             int key )
```

simulates a button click release on the os cursor. The param gives the used key for the click

Parameters

key	

3.8.2.3 getMutex()

```
void mrmOSCursor::getMutex ( )
```

get the mutex. This function will make a blocking wait until you have acces to the mutex

3.8.2.4 releaseMutex()

```
void mrmOSCursor::releaseMutex ( )
```

release the mutex. This function frees the mutex if it is in your control

3.8.2.5 setCursor()

```
void mrmOSCursor::setCursor (
             int posx,
             int posy )
```

sets the cursor position to the parameter values for posx and y.

Parameters

posx	
posy	

3.8.3 Member Data Documentation

3.8.3.1 dpy

```
Display* mrmOSCursor::dpy
```

3.8.3.2 mutex

```
pthread_mutex_t mrmOSCursor::mutex
```

3.8.3.3 posx

```
int mrmOSCursor::posx
```

3.8.3.4 posy

```
int mrmOSCursor::posy
```

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

- inc/mrmOSCursor.h
- src/mrmOSCursor.cpp

3.9 mrmScreen Class Reference

```
#include <mrmScreen.h>
```

Public Member Functions

- mrmScreen (string ip, int pos)
- int getDirection ()
- string getipaddr ()

Public Attributes

- string ip
- int pos

3.9.1 Constructor & Destructor Documentation

3.9.1.1 mrmScreen()

```
mrmScreen::mrmScreen (
              string ip,
              int pos )
```

creates the screen with all needed information, like position and IP

Parameters

ip	
pos	

3.9.2 Member Function Documentation

3.9.2.1 getDirection()

```
int mrmScreen::getDirection ( )
```

getter function to receive the Direction of the screen

Returns

the direction as integer -1 and 1

3.9.2.2 getipaddr()

```
string mrmScreen::getipaddr ( )
```

getter function to receive the IP of the screen

Returns

returns the IP as string value

3.9.3 Member Data Documentation

3.9.3.1 ip

```
string mrmScreen::ip
```

3.9.3.2 pos

```
int mrmScreen::pos
```

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

- inc/mrmScreen.h
- src/mrmScreen.cpp

3.10 mrmUDPClient Class Reference

```
#include <mrmUDPClient.h>
```

Public Member Functions

- mrmUDPClient (int port)
- void sendevent (json sendevent)

Private Attributes

- int s
- struct sockaddr_in addrout

3.10.1 Constructor & Destructor Documentation

3.10.1.1 mrmUDPClient()

```
mrmUDPClient::mrmUDPClient (
              int port )
```

create the UDP client on the parameter port

Parameters

port	

3.10.2 Member Function Documentation

3.10.2.1 sendevent()

```
void mrmUDPClient::sendevent (
             json sendevent )
```

send a event over the network. The event must be given as a json object

Parameters

sendevent	

3.10.3 Member Data Documentation

3.10.3.1 addrout

```
struct sockaddr_in mrmUDPClient::addrout  [private]
```

3.10.3.2 s

```
int mrmUDPClient::s  [private]
```

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

- inc/mrmUDPClient.h
- src/mrmUDPClient.cpp

3.11 mrmUDPServer Class Reference

```
#include <mrmUDPServer.h>
```

Public Member Functions

- mrmUDPServer (int port, string hostip)
- void start ()
- void run ()

Public Attributes

- int s
- struct sockaddr_in addrin
- string hostip

3.11.1 Constructor & Destructor Documentation

3.11.1.1 mrmUDPServer()

```
mrmUDPServer::mrmUDPServer (
             int port,
             string hostip )
```

creating the USP server and will be opened with the parameter port. The hostip is needed to chekc which messages are import for our server

Parameters

port	
hostip	

3.11.2 Member Function Documentation

3.11.2.1 run()

```
void mrmUDPServer::run ( )
```

the run function. This function will bi called from the server thread. The run function receives UDP packages and send them to our controller

3.11.2.2 start()

```
void mrmUDPServer::start ( )
```

starts the Thread which is running the server. The run function wil be called in the thread.

3.11.3 Member Data Documentation

3.11.3.1 addrin

```
struct sockaddr_in mrmUDPServer::addrin
```

3.11.3.2 hostip

```
string mrmUDPServer::hostip
```

3.11.3.3 s

```
int mrmUDPServer::s
```

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

- inc/mrmUDPServer.h
- src/mrmUDPServer.cpp

Chapter 4

File Documentation

4.1 inc/mrmCfgParser.h File Reference

```
#include <string>
#include <list>
#include "mrmMouse.h"
#include "mrmCursor.h"
#include "mrmScreen.h"
#include <nlohmann/json.hpp>
```

Classes

- class mrmCfgParser

Typedefs

- using json = nlohmann::json

4.1.1 Typedef Documentation

4.1.1.1 json

```
using json = nlohmann::json
```

4.2 inc/mrmCursor.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <GL/gl.h>
#include <GL/glx.h>
#include <GL/glxext.h>
#include <X11/Xatom.h>
#include <X11/extensions/Xrender.h>
#include <X11/Xutil.h>
#include <string>
```

Classes

- class mrmCursor

4.3 inc/mrmCursorControl.h File Reference

```
#include <list>
#include "mrmCursor.h"
#include "mrmOSCursor.h"
#include <nlohmann/json.hpp>
```

Classes

- class mrmCursorControl

Typedefs

- using json = nlohmann::json

4.3.1 Typedef Documentation

4.3.1.1 json

```
using json = nlohmann::json
```

4.4 inc/mrmMiceControl.h File Reference

```
#include <list>
#include "../inc/mrmMouse.h"
```

Classes

- class mrmMiceControl

4.5 inc/mrmMouse.h File Reference

```
#include <string>
#include <unistd.h>
#include <iostream>
#include "../inc/osinput.h"
#include <fcntl.h>
```

Classes

- class mrmMouse

4.6 inc/mrmMultiRemoteMouse.h File Reference

```
#include <list>
#include "mrmUDPClient.h"
#include "mrmMouse.h"
#include "mrmCursor.h"
#include "mrmScreen.h"
#include "mrmUDPServer.h"
#include "mrmMiceControl.h"
#include "mrmCursorControl.h"
#include "mrmOSCursor.h"
#include <nlohmann/json.hpp>
```

Classes

- class mrmMultiRemoteMouse

Typedefs

- using json = nlohmann::json

4.6.1 Typedef Documentation

4.6.1.1 json

```
using json = nlohmann::json
```

4.7 inc/mrmOSCursor.h File Reference

```
#include <X11/Xatom.h>
#include <X11/extensions/Xrender.h>
#include <X11/extensions/XTest.h>
#include <X11/Xutil.h>
```

Classes

- class mrmOSCursor

4.8 inc/mrmScreen.h File Reference

```
#include <string>
```

Classes

- class mrmScreen

4.9 inc/mrmUDPClient.h File Reference

```
#include <netinet/in.h>
#include "../inc/mrmCfgParser.h"
```

Classes

- class mrmUDPClient

4.10 inc/mrmUDPServer.h File Reference

```
#include <netinet/in.h>
#include <nlohmann/json.hpp>
```

Classes

- class mrmUDPServer

Typedefs

- using json = nlohmann::json

4.10.1 Typedef Documentation

4.10.1.1 json

```
using json = nlohmann::json
```

4.11 inc/osinput.h File Reference

Classes

- struct input_event

Macros

- #define EV_KEY 0x01
- #define EV_REL 0x02
- #define EV_ABS 0x03
- #define REL_X 0x00
- #define REL_Y 0x01
- #define REL_WHEEL 0x08
- #define ABS_X 0x00
- #define ABS_Y 0x01
- #define ABS_WHEEL 0x08
- #define BTN_LEFT 0x110
- #define BTN_RIGHT 0x111
- #define BTN_MIDDLE 0x112

4.11.1 Macro Definition Documentation

4.11.1.1 ABS_WHEEL

```
#define ABS_WHEEL 0x08
```

4.11.1.2 ABS_X

```
#define ABS_X 0x00
```

4.11.1.3 ABS_Y

```
#define ABS_Y 0x01
```

4.11.1.4 BTN_LEFT

```
#define BTN_LEFT 0x110
```

4.11.1.5 BTN_MIDDLE

```
#define BTN_MIDDLE 0x112
```

4.11.1.6 BTN_RIGHT

```
#define BTN_RIGHT 0x111
```

4.11.1.7 EV_ABS

```
#define EV_ABS 0x03
```

4.11.1.8 EV_KEY

```
#define EV_KEY 0x01
```

4.11.1.9 EV_REL

```
#define EV_REL 0x02
```

4.11.1.10 REL_WHEEL

```
#define REL_WHEEL 0x08
```

4.11.1.11 REL_X

```
#define REL_X 0x00
```

4.11.1.12 REL_Y

```
#define REL_Y 0x01
```

4.12 src/main.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <sstream>
#include "../inc/mrmCfgParser.h"
#include "../inc/mrmMultiRemoteMouse.h"
```

Functions

- int main (int argc, char **argv)

Variables

- mrmMultiRemoteMouse * mrm = nullptr
- string cfgfilename = "../cfg/config.json"

4.12.1 Function Documentation

4.12.1.1 main()

```
int main (
             int argc,
             char ** argv )
```

4.12.2 Variable Documentation

4.12.2.1 cfgfilename

```
string cfgfilename = "../cfg/config.json"
```

4.12.2.2 mrm

```
mrmMultiRemoteMouse* mrm = nullptr
```

4.13 src/mrmCfgParser.cpp File Reference

```
#include <nlohmann/json.hpp>
#include <iostream>
#include <list>
#include "../inc/mrmCfgParser.h"
```

4.14 src/mrmCursor.cpp File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <GL/glew.h>
#include <GL/glx.h>
#include <X11/Xatom.h>
#include <X11/extensions/shape.h>
#include <X11/extensions/Xrender.h>
#include <X11/extensions/Xfixes.h>
#include <X11/extensions/Xcomposite.h>
#include <iostream>
#include <sstream>
#include "../inc/mrmCursor.h"
```

Functions

- void ∗ ThreadCursor (void ∗pVoid)

4.14.1 Function Documentation

4.14.1.1 ThreadCursor()

```
void* ThreadCursor (
            void * pVoid )
```

4.15 src/mrmCursorControl.cpp File Reference

```
#include <iostream>
#include "../inc/osinput.h"
#include "../inc/mrmOSCursor.h"
#include "../inc/mrmCursorControl.h"
#include "../inc/mrmMultiRemoteMouse.h"
```

Variables

- mrmMultiRemoteMouse ∗ mrm

4.15.1 Variable Documentation

4.15.1.1 mrm

```
mrmMultiRemoteMouse* mrm
```

4.16 src/mrmMiceControl.cpp File Reference

```
#include <utility>
#include "../inc/mrmMiceControl.h"
```

4.17 src/mrmMouse.cpp File Reference

```
#include <unistd.h>
#include <iostream>
#include "../inc/osinput.h"
#include <fcntl.h>
#include <cstring>
#include "../inc/mrmMouse.h"
#include "../inc/mrmMultiRemoteMouse.h"
```

Functions

- void ∗ ThreadMouse (void ∗pVoid)

Variables

- mrmMultiRemoteMouse ∗ mrm

4.17.1 Function Documentation

4.17.1.1 ThreadMouse()

```
void* ThreadMouse (
            void * pVoid )
```

4.17.2 Variable Documentation

4.17.2.1 mrm

```
mrmMultiRemoteMouse* mrm
```

4.18 src/mrmMultiRemoteMouse.cpp File Reference

```
#include "../inc/mrmMultiRemoteMouse.h"
```

4.19 src/mrmOSCursor.cpp File Reference

```
#include <cstring>
#include <iostream>
#include "../inc/mrmOSCursor.h"
```

4.20 src/mrmScreen.cpp File Reference

```
#include "../inc/mrmScreen.h"
```

4.21 src/mrmUDPClient.cpp File Reference

```
#include <iostream>
#include <cstring>
#include "../inc/mrmUDPClient.h"
```

4.22 src/mrmUDPServer.cpp File Reference

```
#include <iostream>
#include <arpa/inet.h>
#include "../inc/mrmUDPServer.h"
#include "../inc/mrmMultiRemoteMouse.h"
```

Functions

- void ∗ ThreadUDPServer (void ∗pVoid)

Variables

- mrmMultiRemoteMouse ∗ mrm

4.22.1 Function Documentation

4.22.1.1 ThreadUDPServer()

```
void* ThreadUDPServer (
            void * pVoid )
```

4.22.2 Variable Documentation

4.22.2.1 mrm

```
mrmMultiRemoteMouse* mrm
```

Index