Graphics

COLLABORATORS

	TITLE : Graphics	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 6, 2016	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

List of Figures

Chapter 1

Data Structure Documentation

1.1 Canvas struct Reference

```
#include <canvas.h>
```

Inheritance diagram for Canvas

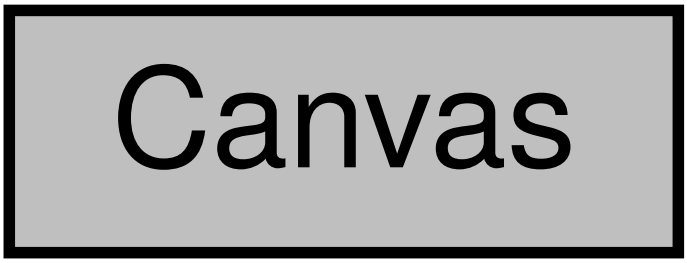

Figure 1.1: Inheritance graph

Collaboration diagram for Canvas

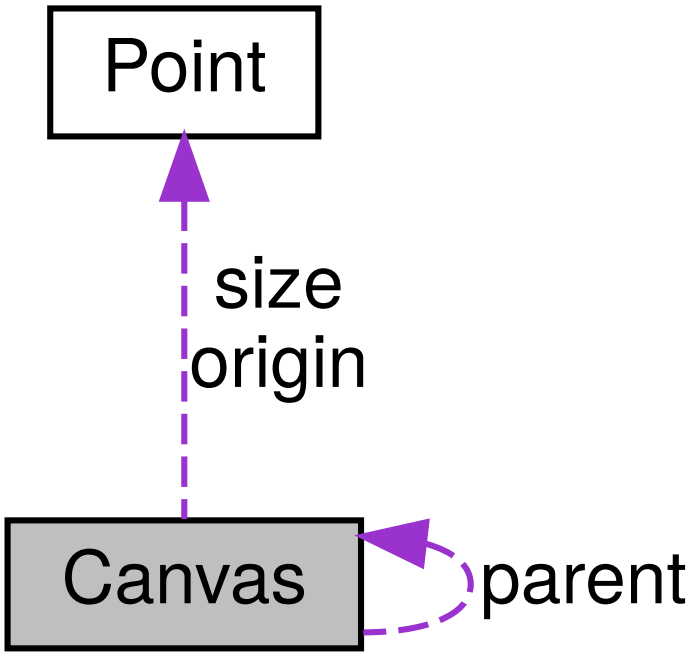

Figure 1.2: Collaboration graph

1.1.1 Data Fields

- SDL_Surface * surface
- Point size
- Point origin
- struct Canvas * parent

1.1.2 A

Canvas is part of a Window or of another Canvas, on which it's possible to draw.

1.1.3 Field Documentation

1.1.3.1 SDL_Surface* Canvas::surface

Pointer to the SDL_Surface used to store the content of the Canvas, user shouldn't have to touch this.
Definition at line 17 of file canvas.h
The Documentation for this struct was generated from the following file:

- canvas.h

1.1.3.2 Point Canvas::size

Point representing the size of the Canvas, usefull to get the value quickly, but user souldn't change it.
Definition at line 18 of file canvas.h
The Documentation for this struct was generated from the following file:

- canvas.h

1.1.3.3 Point Canvas::origin

Point representing the origin of the Canvas, user can set and get it safely.

Definition at line 19 of file canvas.h

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

- canvas.h

1.1.3.4 struct Canvas* Canvas::parent

Pointer to the Canvas representing the parent of the Canvas, i.e. the one one which it will be blitted, if the Canvas is the root Canvas representing the whole Window it points to NULL.

Definition at line 20 of file canvas.h

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

- canvas.h

1.2 Circle struct Reference

```
#include <circle.h>
```

Inheritance diagram for Circle

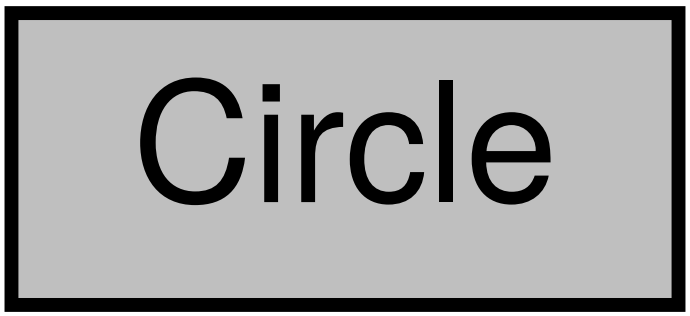

Figure 1.3: Inheritance graph

Collaboration diagram for Circle

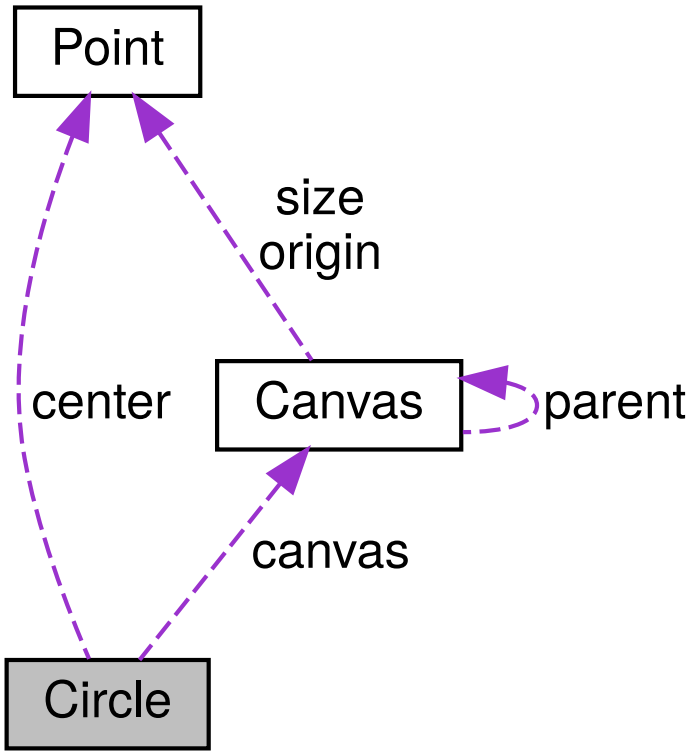

Figure 1.4: Collaboration graph

1.2.1 Data Fields

- Point center
- int radius
- Canvas * canvas

1.2.2 Field Documentation

1.3 Color struct Reference

```
#include <color.h>
```

Inheritance diagram for Color

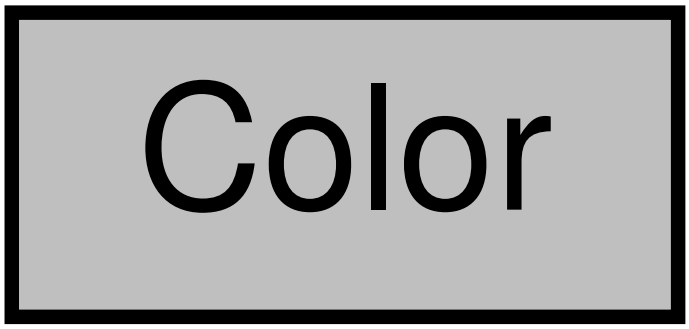

Figure 1.5: Inheritance graph

Collaboration diagram for Color

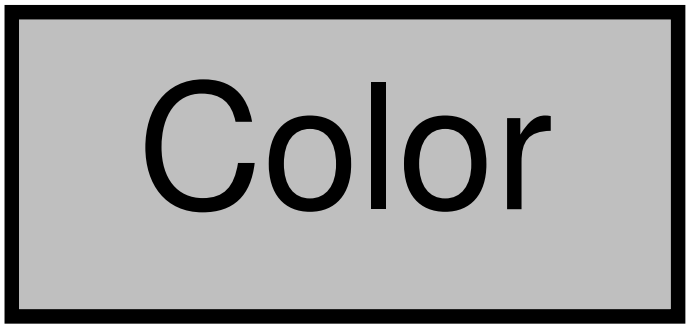

Figure 1.6: Collaboration graph

1.3.1 Data Fields

- Uint32 rgb
- Uint8 alpha

1.3.2 Field Documentation

1.4 Event struct Reference

```
#include <event.h>
```

Inheritance diagram for Event

Figure 1.7: Inheritance graph

Collaboration diagram for Event

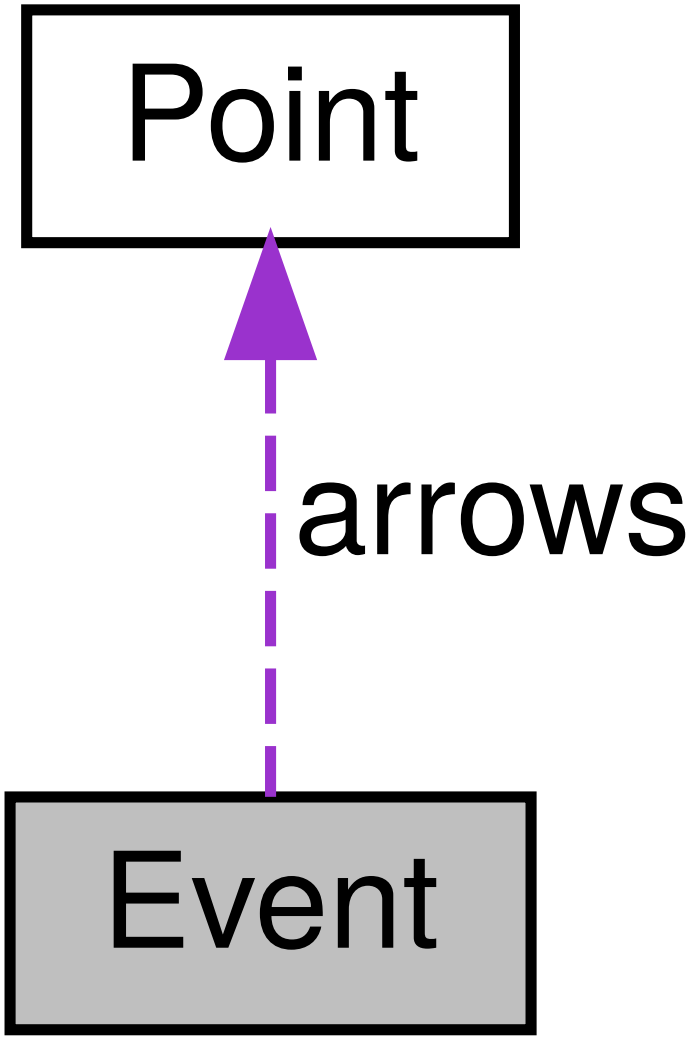

Figure 1.8: Collaboration graph

1.4.1 Data Fields

- bool quit

- bool space
- Point arrows

1.4.2 Field Documentation

1.5 Image struct Reference

```
#include <image.h>
```

Inheritance diagram for Image

Figure 1.9: Inheritance graph

Collaboration diagram for Image

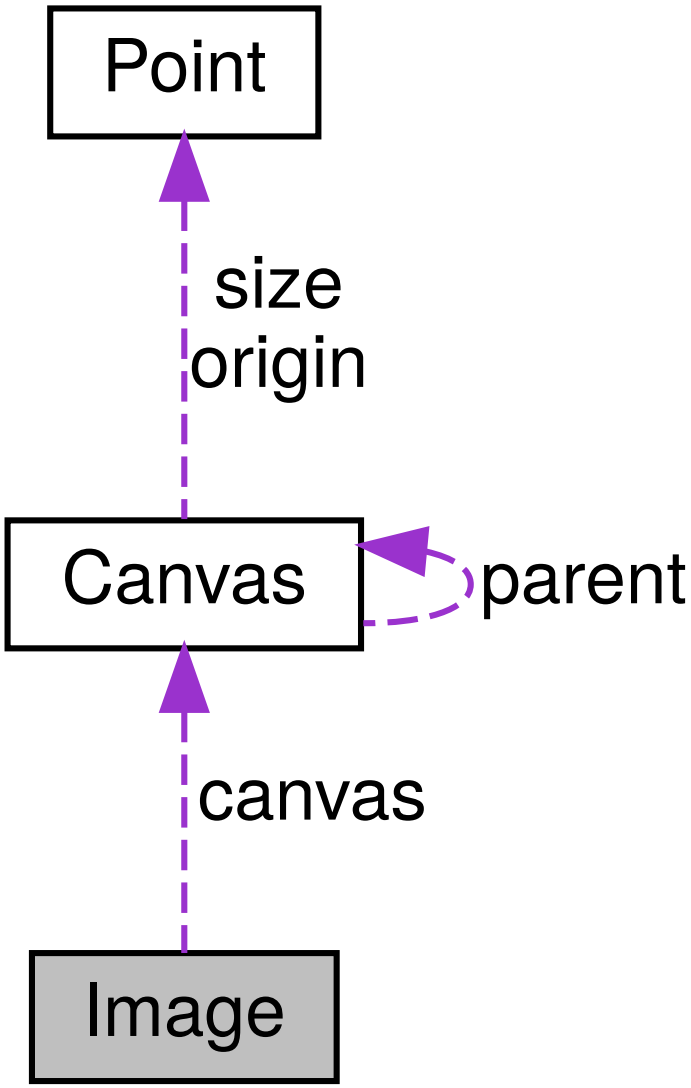

Figure 1.10: Collaboration graph

1.5.1 Data Fields

- SDL_Surface * surface
- Canvas * canvas

1.5.2 Field Documentation

1.6 Line struct Reference

```
#include <line.h>
```

Inheritance diagram for Line

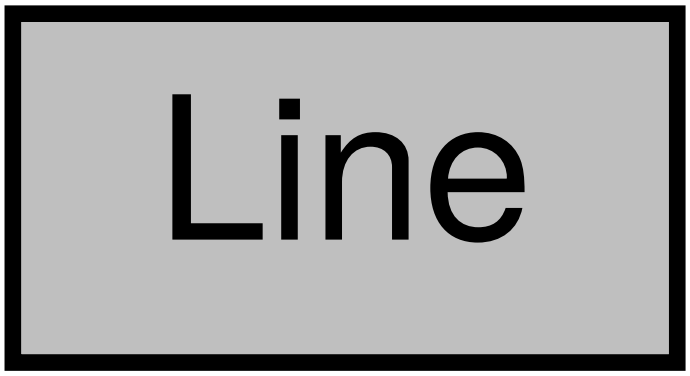

Figure 1.11: Inheritance graph

Collaboration diagram for Line

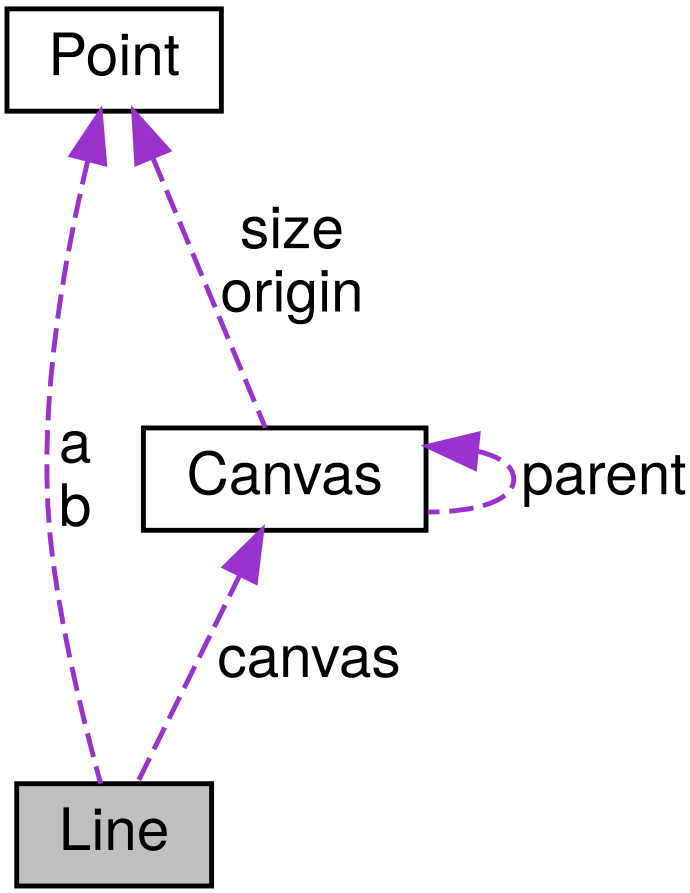

Figure 1.12: Collaboration graph

1.6.1 Data Fields

- Point a
- Point b
- Canvas * canvas

1.6.2 Field Documentation

1.7 Pixel struct Reference

```
#include <pixel.h>
```

Inheritance diagram for Pixel

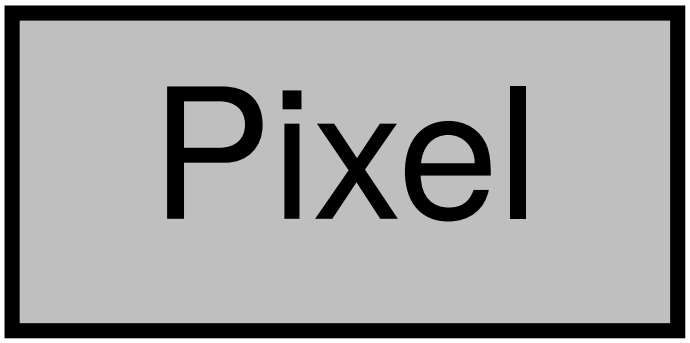

Figure 1.13: Inheritance graph

Collaboration diagram for Pixel

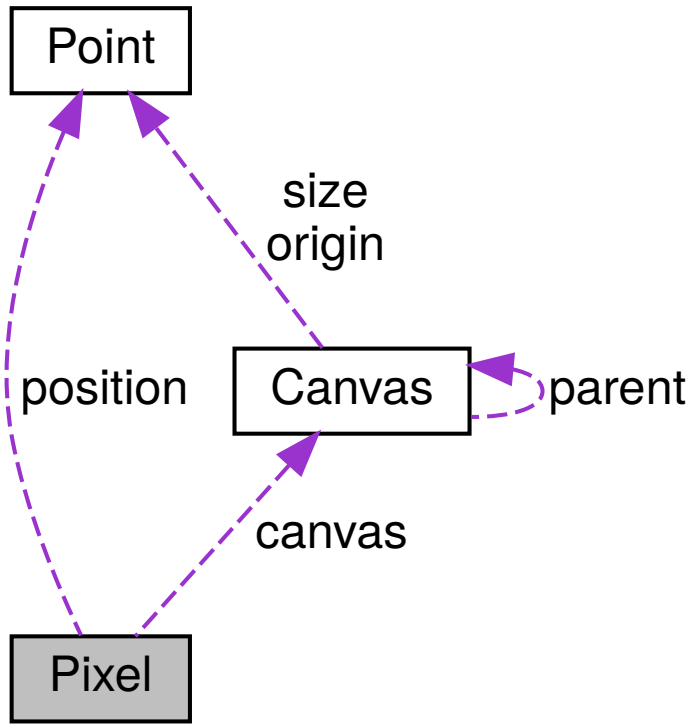

Figure 1.14: Collaboration graph

1.7.1 Data Fields

- Point position
- Canvas * canvas

1.7.2 Field Documentation

1.8 Point struct Reference

```
#include <point.h>
```

Inheritance diagram for Point

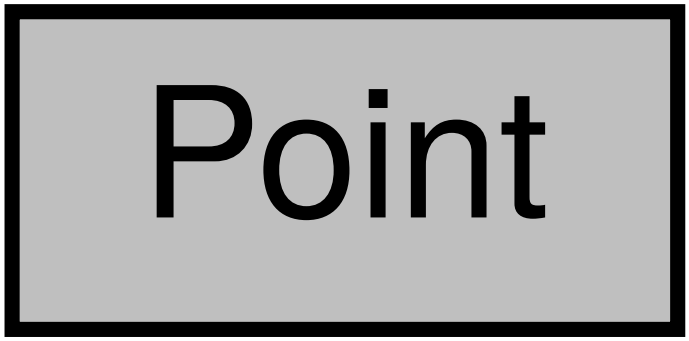

Figure 1.15: Inheritance graph

Collaboration diagram for Point

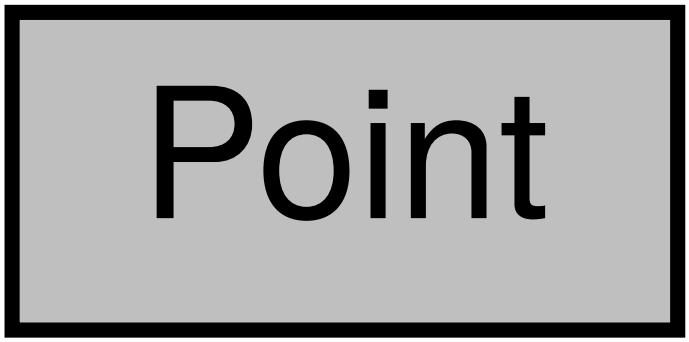

Figure 1.16: Collaboration graph

1.8.1 Data Fields

- int x
- int y

1.8.2 Field Documentation

1.9 Rectangle struct Reference

```
#include <rectangle.h>
```

Inheritance diagram for Rectangle

Figure 1.17: Inheritance graph

Collaboration diagram for Rectangle

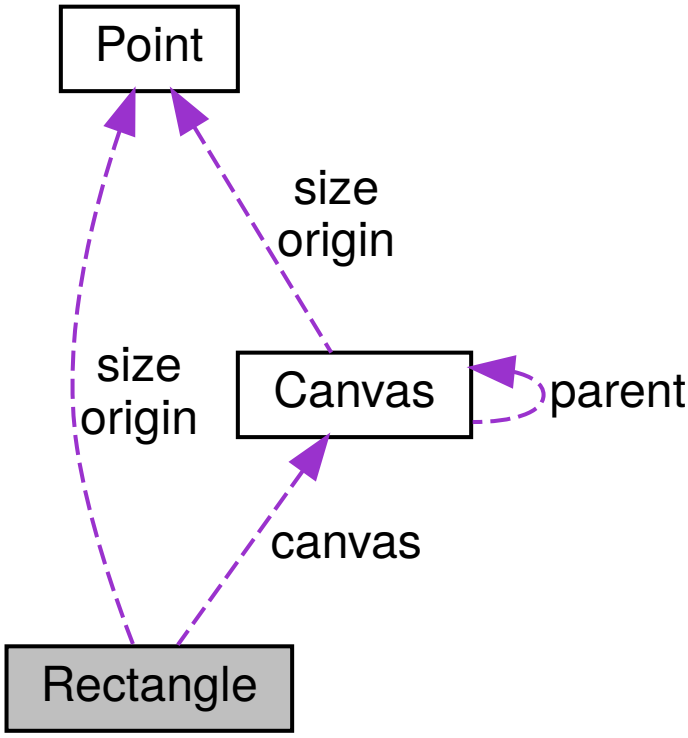

Figure 1.18: Collaboration graph

1.9.1 Data Fields

- Point origin
- Point size
- Canvas * canvas

1.9.2 Field Documentation

1.10 Sound struct Reference

```
#include <sound.h>
```

Inheritance diagram for Sound

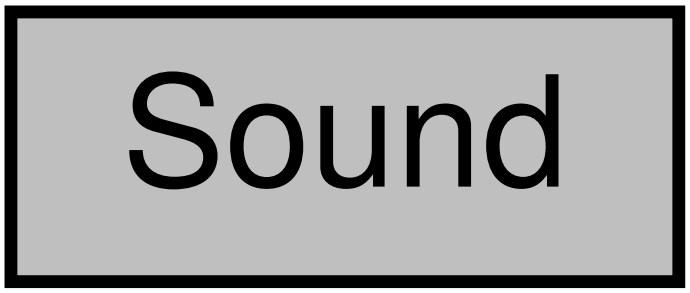

Figure 1.19: Inheritance graph

Collaboration diagram for Sound

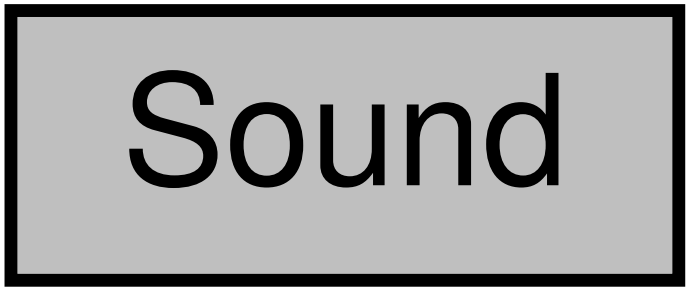

Figure 1.20: Collaboration graph

1.10.1 Data Fields

- Mix_Music * content

1.10.2 Field Documentation

1.11 Sphere struct Reference

```
#include <sphere.h>
```

Inheritance diagram for Sphere

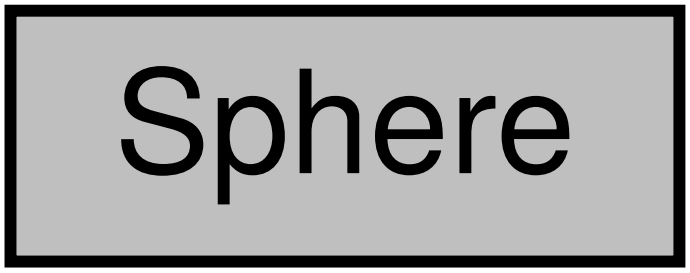

Figure 1.21: Inheritance graph

Collaboration diagram for Sphere

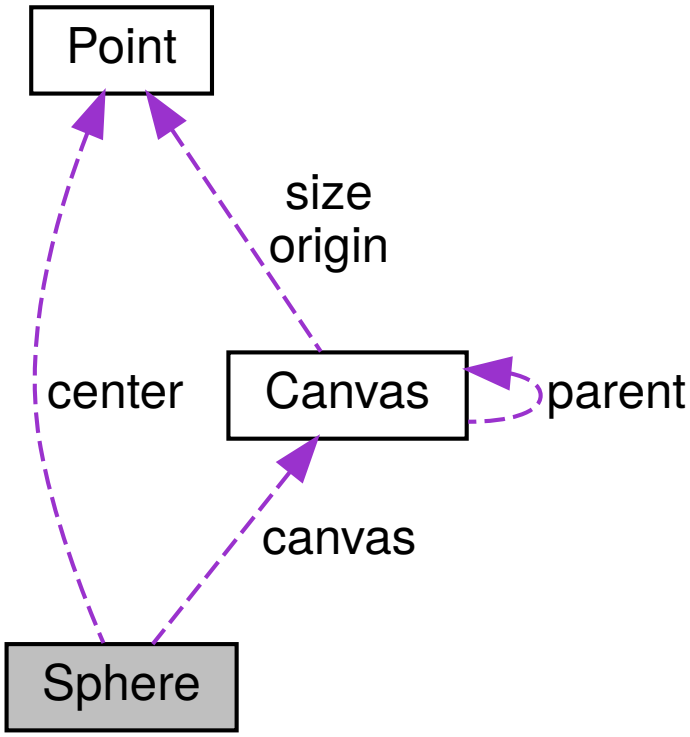

Figure 1.22: Collaboration graph

1.11.1 Data Fields

- Point center
- int radius
- Canvas * canvas

1.11.2 Field Documentation

1.12 Window struct Reference

```
#include <window.h>
```

Inheritance diagram for Window

Figure 1.23: Inheritance graph

Collaboration diagram for Window

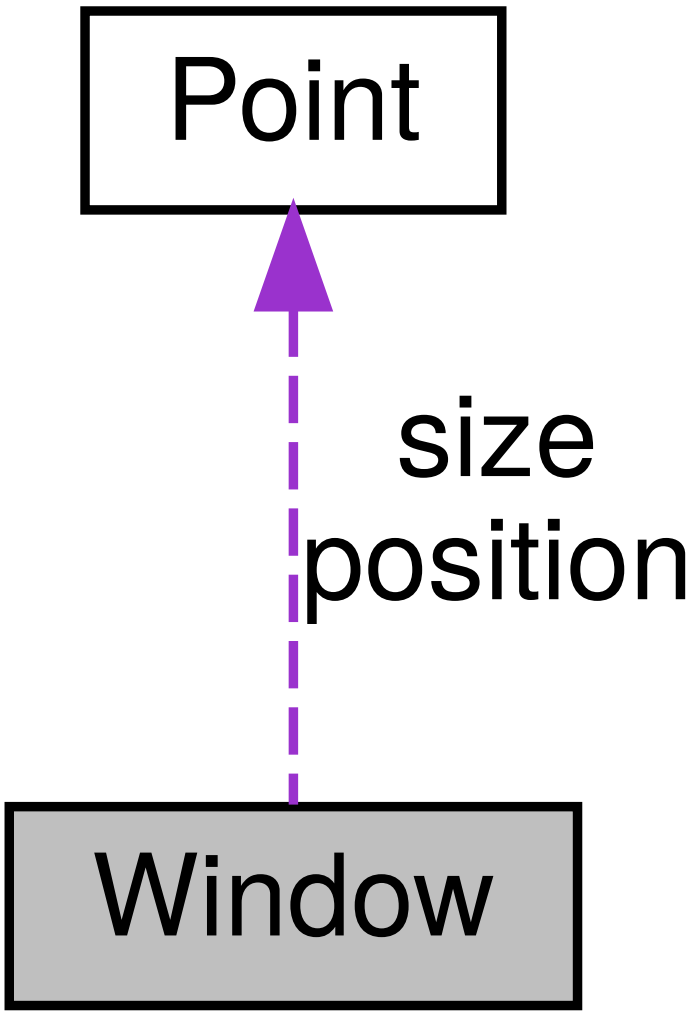

Figure 1.24: Collaboration graph

1.12.1 Data Fields

- SDL_Window * window
- char * title
- Point position
- Point size

1.12.2 Field Documentation

Chapter 2

File Documentation

2.1 calc.h File Reference

```
#include <stdlib.h>
```

```
#include <unistd.h>
```

Include dependency diagram for calc.h

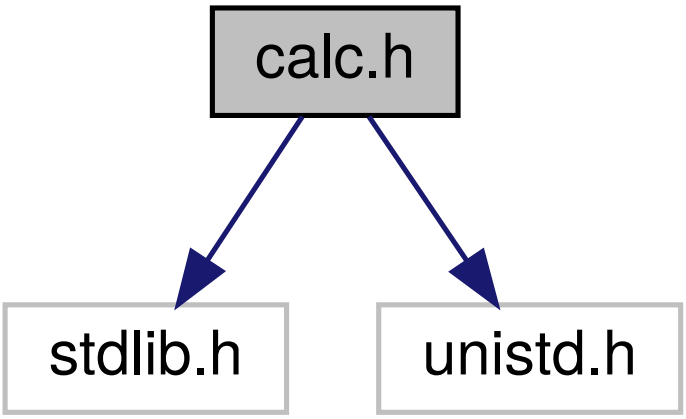

Figure 2.1: Dependency diagram

Included by dependency diagram for calc.h

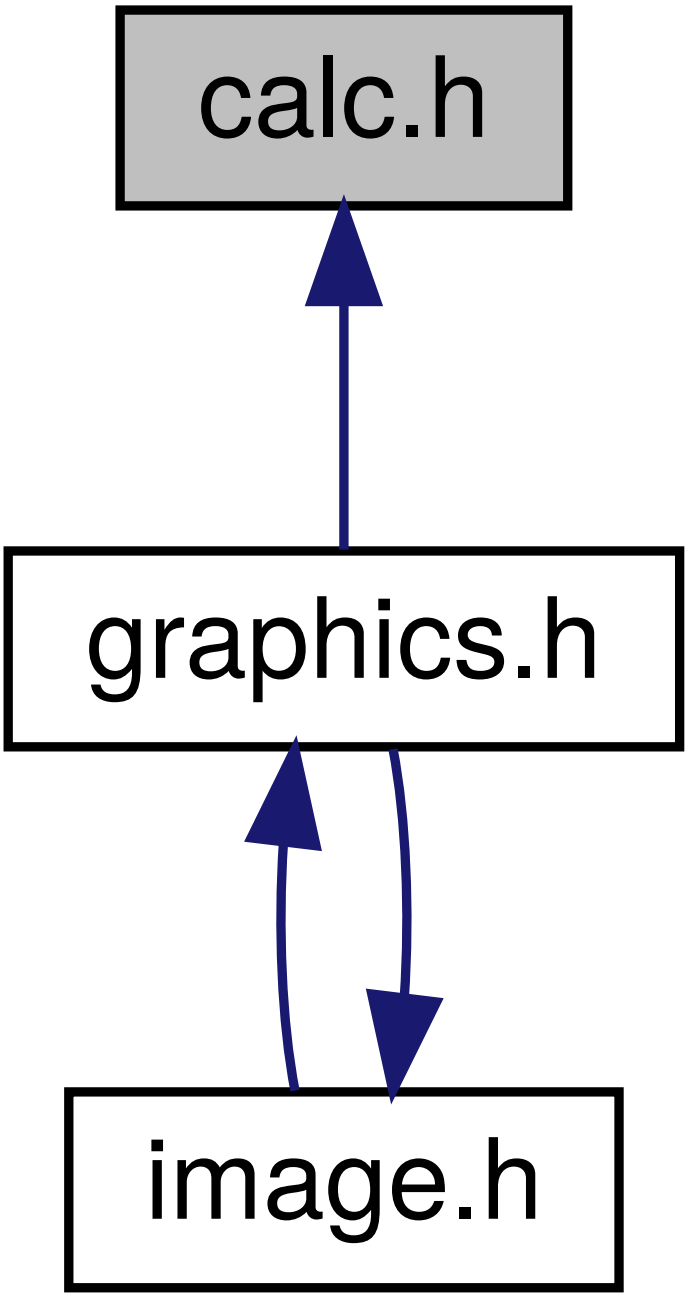

Figure 2.2: Dependency diagram

2.1.1 Functions

- float calc_alea_float (void) *Function to get a random float x in [0 ; 1[.*
- int calc_alea_int (const int min, const int max) *Function to get a random int.*

2.1.2 Detailed Description

Some maths functions.

Definition in file calc.h

```
1
6 #ifndef DEF_CALC_H
7 #define DEF_CALC_H
8
9 #include <stdlib.h>
10 #include <unistd.h>
11
17 float calc_alea_float(void);
18
26 int calc_alea_int(const int min, const int max);
27
28 #endif
```

2.2 canvas.h File Reference

```
#include "window.h"
```

```
#include "color.h"
```

```
#include "rectangle.h"
```

Include dependency diagram for canvas.h

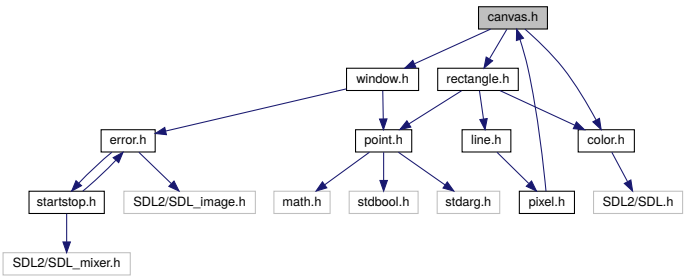

Figure 2.3: Dependency diagram

Included by dependency diagram for canvas.h

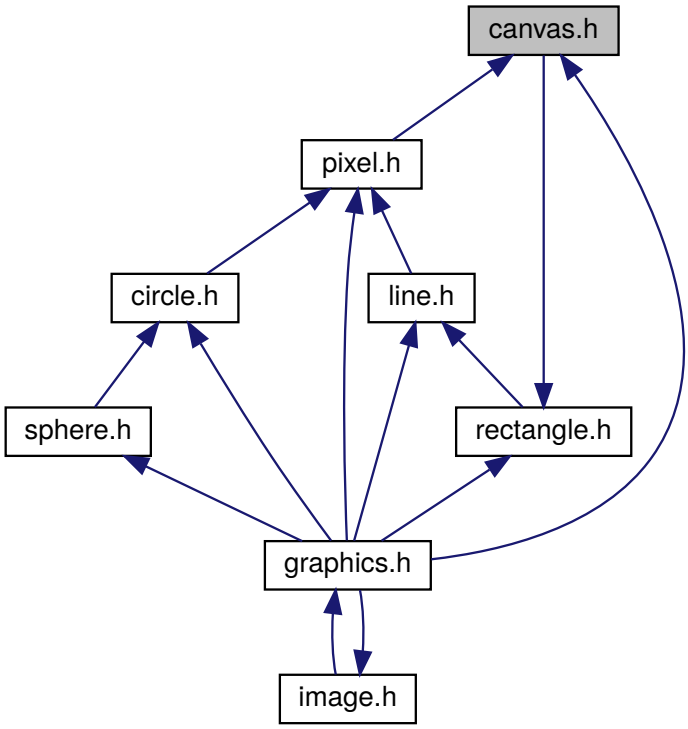

Figure 2.4: Dependency diagram

2.2.1 Data Structures

- struct Canvas

  A Canvas is part of a Window or of another Canvas, on which it's possible to draw.

2.2.2 Typedefs

- typedef struct Canvas Canvas

2.2.3 Functions

- bool canvas_collision_canvas (const Canvas * canvas1, const Canvas * canvas2) *Function to detect collision between two Canvas.*
- bool canvas_is_out_of_parent_bottom (const Canvas * canvas) *Function to know if a Canvas is under its parent.*
- bool canvas_is_out_of_parent_left (const Canvas * canvas) *Function to know if a Canvas is out of its parent's left side.*
- bool canvas_is_out_of_parent_right (const Canvas * canvas) *Function to know if a Canvas is out of its parent's right side.*
- bool canvas_is_out_of_parent_top (const Canvas * canvas) *Function to know if a Canvas is upper its parent's.*
- bool canvas_is_out_of_parent_x (const Canvas * canvas) *Function to know if a Canvas is outside of its parent's on the X axis.*
- bool canvas_is_out_of_parent_y (const Canvas * canvas) *Function to know if a Canvas is outside of its parent's on the Y axis.*

- bool canvas_will_be_out_of_parent_bottom (const Canvas * canvas, const Point * move) *Function to know if a Canvas will be under its parent after moving its origin.*
- bool canvas_will_be_out_of_parent_left (const Canvas * canvas, const Point * move) *Function to know if a Canvas will be out of its parent's left side after moving its origin.*
- bool canvas_will_be_out_of_parent_right (const Canvas * canvas, const Point * move) *Function to know if a Canvas will be out of its parent's right side after moving its origin.*
- bool canvas_will_be_out_of_parent_top (const Canvas * canvas, const Point * move) *Function to know if a Canvas will be upper its parent after moving its origin.*
- bool canvas_will_be_out_of_parent_x (const Canvas * canvas, const Point * move) *Function to know if a Canvas will be outside of its parent on the X axis after moving its origin.*
- bool canvas_will_be_out_of_parent_y (const Canvas * canvas, const Point * move) *Function to know if a Canvas will be outside of its parent on the Y axis after moving its origin.*
- void canvas_blit (Canvas * canvas) *Function to blit a Canvas on its parent.*
- void canvas_create (Canvas * canvas, const Point * size, const Point * origin, Canvas * parent) *Function to create a Canvas.*
- void canvas_clear (Canvas * canvas) *Function to clear a Canvas, i.e. filling it with black.*
- void canvas_create_from_window (Canvas * canvas, const Window * window) *Function to create a Canvas from a Window, it will fill the whole window.*
- void canvas_draw_borders_in (Canvas * canvas, const Color * color) *Function to draw a 1 pixel border inside of a Canvas.*
- void canvas_draw_borders_out (Canvas * canvas, const Color * color) *Function to draw a 1 pixel border outside of a Canvas.*
- void canvas_fill (Canvas * canvas, const Color * color) *Function to fill a Canvas with a Color.*
- void canvas_get_absolute_origin (const Canvas * canvas, Point * absoluteOrigin) *Function to get the origin of a Canvas on the Window, instead of on its parent.*

2.2.4 Detailed Description

All functions related to Canvas.

Definition in file canvas.h

```
1
6 #ifndef DEF_CANVAS_H
7 #define DEF_CANVAS_H
8
9 #include "window.h"
10 #include "color.h"
11
16 typedef struct Canvas {
17     SDL_Surface* surface;
18     Point size;
19     Point origin;
20     struct Canvas* parent;
```

```
21 } Canvas;
22
23 #include "rectangle.h"
24
33 bool canvas_collision_canvas(const Canvas* canvas1, const Canvas* canvas2)  ↩
   __attribute__((pure));
34
42 bool canvas_is_out_of_parent_bottom(const Canvas* canvas) __attribute__((pure)) ↩
   ;
43
51 bool canvas_is_out_of_parent_left(const Canvas* canvas) __attribute__((pure));
52
60 bool canvas_is_out_of_parent_right(const Canvas* canvas) __attribute__((pure));
61
69 bool canvas_is_out_of_parent_top(const Canvas* canvas) __attribute__((pure));
70
78 bool canvas_is_out_of_parent_x(const Canvas* canvas) __attribute__((pure));
79
87 bool canvas_is_out_of_parent_y(const Canvas* canvas) __attribute__((pure));
88
97 bool canvas_will_be_out_of_parent_bottom(const Canvas* canvas, const Point*  ↩
   move) __attribute__((pure));
98
107 bool canvas_will_be_out_of_parent_left(const Canvas* canvas, const Point* move ↩
   ) __attribute__((pure));
108
117 bool canvas_will_be_out_of_parent_right(const Canvas* canvas, const Point*  ↩
   move) __attribute__((pure));
118
127 bool canvas_will_be_out_of_parent_top(const Canvas* canvas, const Point* move) ↩
    __attribute__((pure));
128
137 bool canvas_will_be_out_of_parent_x(const Canvas* canvas, const Point* move)  ↩
   __attribute__((pure));
138
147 bool canvas_will_be_out_of_parent_y(const Canvas* canvas, const Point* move)  ↩
   __attribute__((pure));
148
155 void canvas_blit(Canvas* canvas);
156
166 void canvas_create(Canvas* canvas, const Point* size, const Point* origin,  ↩
   Canvas* parent);
167
174 void canvas_clear(Canvas* canvas);
175
183 void canvas_create_from_window(Canvas* canvas, const Window* window);
184
192 void canvas_draw_borders_in(Canvas* canvas, const Color* color);
193
201 void canvas_draw_borders_out(Canvas* canvas, const Color* color);
202
210 void canvas_fill(Canvas* canvas, const Color* color);
211
219 void canvas_get_absolute_origin(const Canvas* canvas, Point* absoluteOrigin);
220
221 #endif
```

2.3 circle.h File Reference

```
#include "pixel.h"
```

Include dependency diagram for circle.h

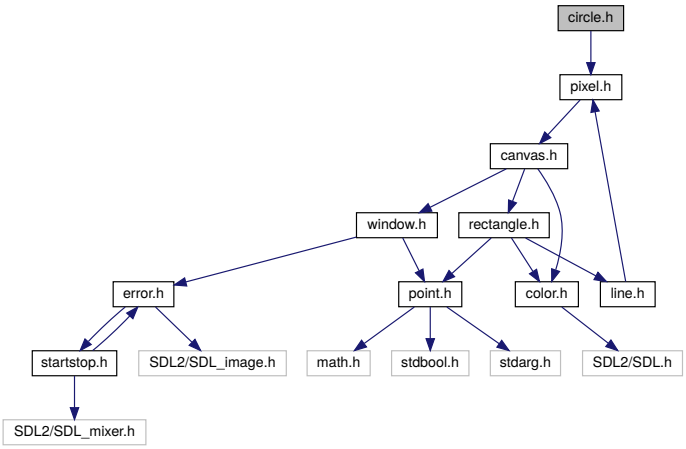

Figure 2.5: Dependency diagram

Included by dependency diagram for circle.h

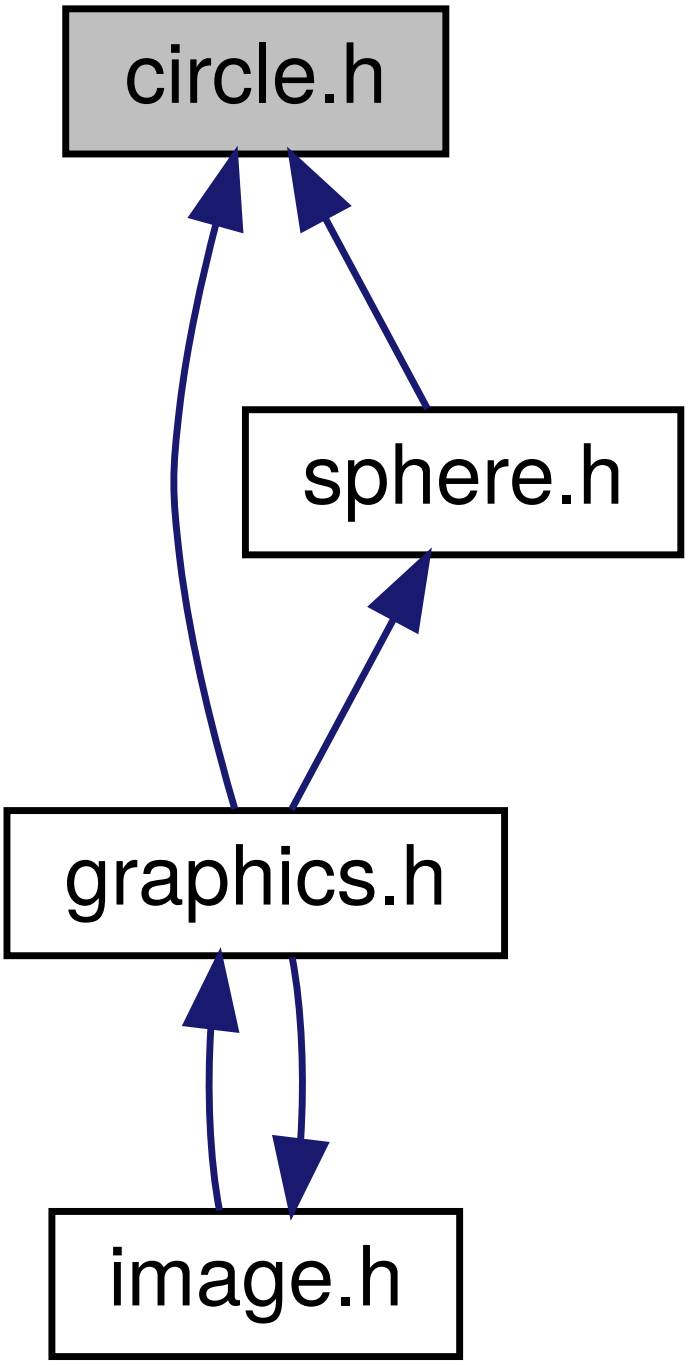

Figure 2.6: Dependency diagram

2.3.1 Data Structures

- struct Circle

2.3.2 Functions

- void circle_draw (const Circle * circle, const Color * color)
- void circle_draw_fill (const Circle * circle, const Color * color)

2.3.3 Detailed Description

Definition in file circle.h

```
1 #ifndef DEF_CIRCLE_H
2 #define DEF_CIRCLE_H
3
4 #include "pixel.h"
5
6 #pragma pack(push, 1)
7 typedef struct {
8     Point center;
9     int radius;
10     Canvas* canvas;
11 } Circle;
12 #pragma pack(pop)
13
14 void circle_draw(const Circle* circle, const Color* color);
15
16 void circle_draw_fill(const Circle* circle, const Color* color);
17
18 #endif
```

2.4 color.h File Reference

```
#include <SDL2/SDL.h>
```

Include dependency diagram for color.h

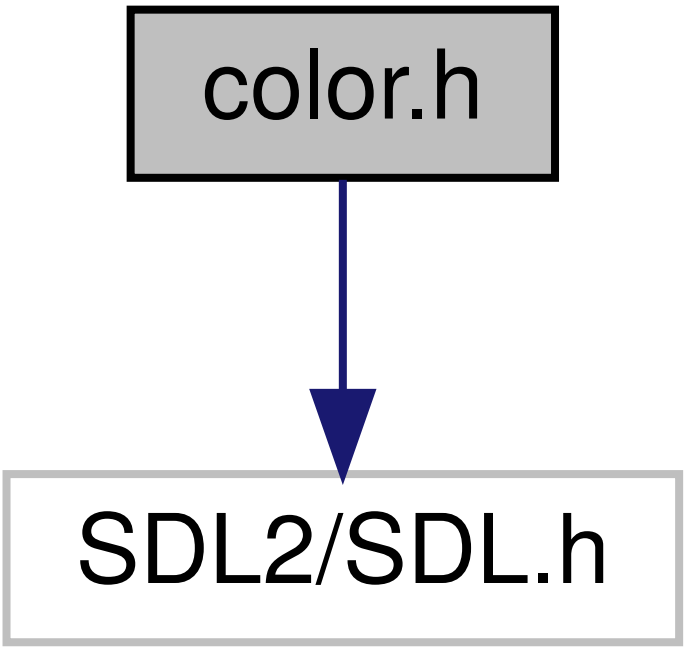

Figure 2.7: Dependency diagram

Included by dependency diagram for color.h

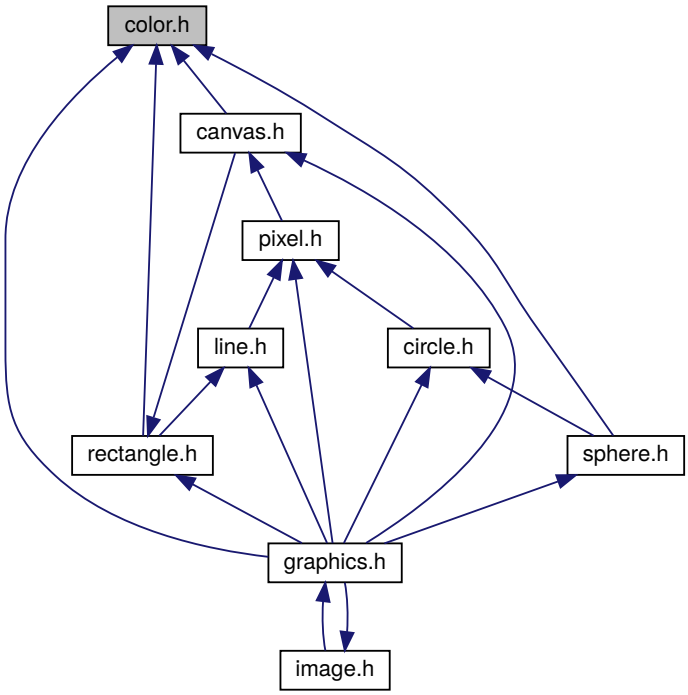

Figure 2.8: Dependency diagram

2.4.1 Data Structures

- struct Color

2.4.2 Functions

- void color_translate (const Color * color, SDL_Color * sdlColor)
- Uint8 color_get_red (const Color * color)
- Uint8 color_get_green (const Color * color)
- Uint8 color_get_blue (const Color * color)

2.4.3 Detailed Description

Definition in file color.h

```
1 #ifndef DEF_COLOR_H
2 #define DEF_COLOR_H
3
4 #include <SDL2/SDL.h>
5
6 #pragma pack(push, 1)
7 typedef struct {
8     Uint32 rgb;
9     Uint8 alpha;
10 } Color;
```

```
11 #pragma pack(pop)
12
13 void color_translate(const Color* color, SDL_Color* sdlColor);
14
15 Uint8 color_get_red(const Color* color) __attribute__((const));
16
17 Uint8 color_get_green(const Color* color) __attribute__((const));
18
19 Uint8 color_get_blue(const Color* color) __attribute__((pure));
20
21 #endif
```

2.5 error.h File Reference

```
#include <SDL2/SDL_image.h>
```

```
#include "startstop.h"
```

Include dependency diagram for error.h

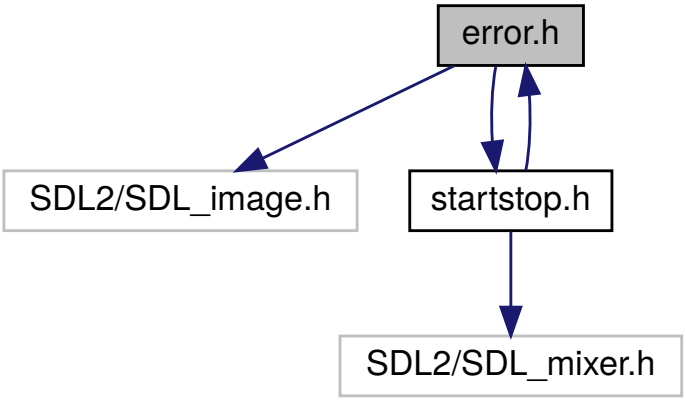

Figure 2.9: Dependency diagram

Included by dependency diagram for error.h

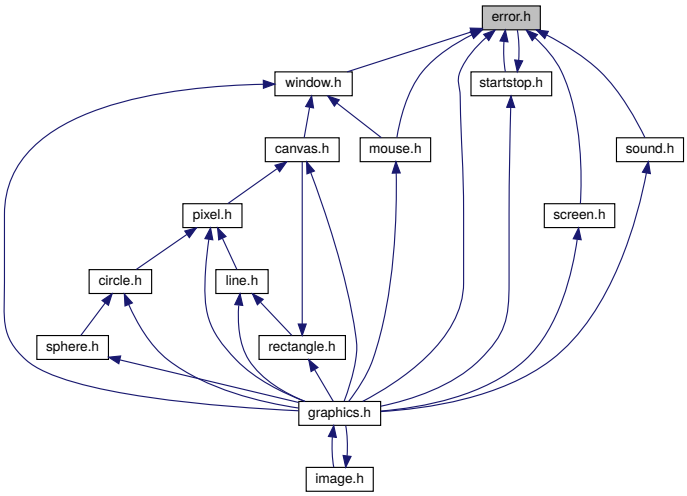

Figure 2.10: Dependency diagram

2.5.1 Functions

- void error_quit (void)

2.5.2 Detailed Description

Definition in file error.h

```
1 #ifndef DEF_ERROR_H
2 #define DEF_ERROR_H
3
4 #include <SDL2/SDL_image.h>
5 #include "startstop.h"
6
7 void error_quit(void) __attribute__ ((noreturn));
8
9 #endif
```

2.6 event.h File Reference

```
#include <SDL2/SDL.h>
```

```
#include "point.h"
```

Include dependency diagram for event.h

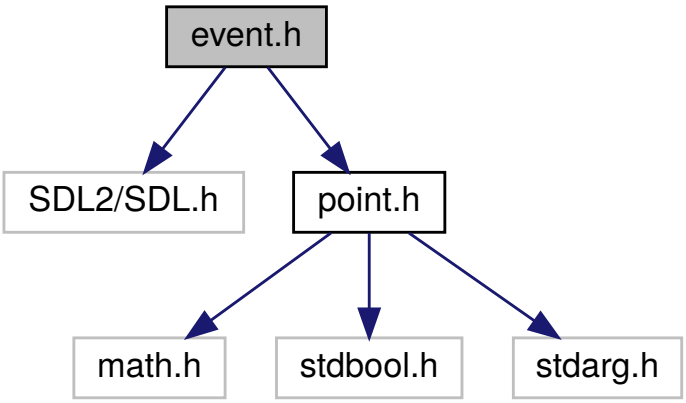

Figure 2.11: Dependency diagram

Included by dependency diagram for event.h

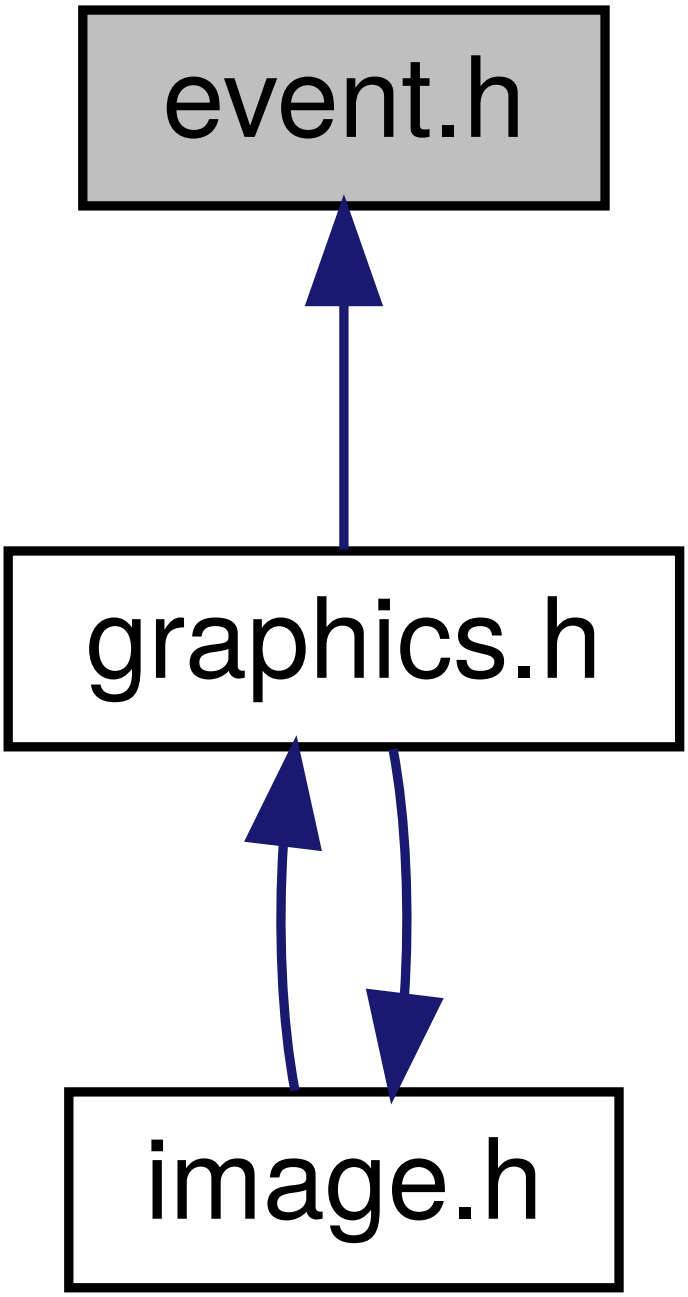

Figure 2.12: Dependency diagram

2.6.1 Data Structures

- struct Event

2.6.2 Functions

- void event_create (Event * newEvent)
- void event_update (Event * event)

2.6.3 Detailed Description

Definition in file event.h

```
1 #ifndef DEF_EVENT_H
2 #define DEF_EVENT_H
3
4 #include <SDL2/SDL.h>
5 #include "point.h"
6
7 #pragma pack(push, 1)
8 typedef struct {
9     bool quit;
10     bool space;
11     Point arrows;
12 } Event;
13 #pragma pack(pop)
14
15 void event_create(Event* newEvent);
16
17 void event_update(Event* event);
18
19 #endif
```

2.7 graphics.h File Reference

```
#include <stdarg.h>
```

```
#include <stdbool.h>
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <unistd.h>
```

```
#include <SDL2/SDL.h>
```

```
#include <SDL2/SDL_image.h>
```

```
#include <SDL2/SDL_mixer.h>
```

```
#include "point.h"
```

```
#include "pixel.h"
```

```
#include "canvas.h"
```

```
#include "line.h"
```

```
#include "window.h"
```

```
#include "screen.h"
```

```
#include "color.h"
```

```
#include "circle.h"
```

```
#include "sound.h"
```

```
#include "calc.h"
```

```
#include "rectangle.h"
```

```
#include "event.h"
```

```
#include "sphere.h"
```

```
#include "image.h"
```

```
#include "error.h"
```

```
#include "startstop.h"
```

```
#include "mouse.h"
```

Include dependency diagram for graphics.h

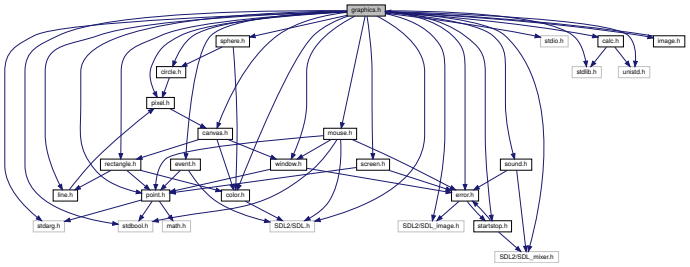

Figure 2.13: Dependency diagram

Included by dependency diagram for graphics.h

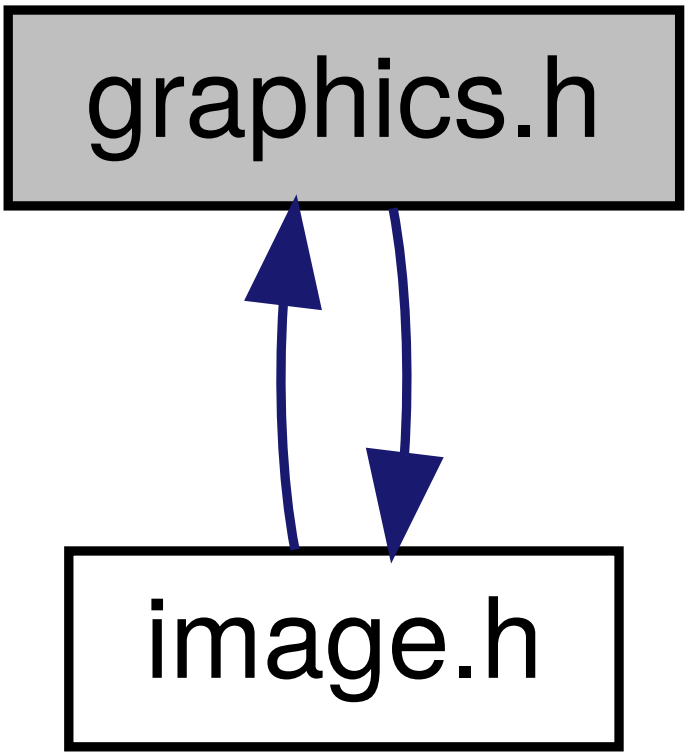

Figure 2.14: Dependency diagram

2.7.1 Detailed Description

Definition in file graphics.h

```
1 #ifndef DEF_GRAPHICS_H
2 #define DEF_GRAPHICS_H
3
4 #include <stdarg.h>
5 #include <stdbool.h>
6 #include <stdio.h>
7 #include <stdlib.h>
8 #include <unistd.h>
9
10 #include <SDL2/SDL.h>
11 #include <SDL2/SDL_image.h>
12 #include <SDL2/SDL_mixer.h>
13
14 #include "point.h"
15 #include "pixel.h"
16 #include "canvas.h"
17 #include "line.h"
18 #include "window.h"
19 #include "screen.h"
20 #include "color.h"
21 #include "circle.h"
22 #include "sound.h"
23 #include "calc.h"
24 #include "rectangle.h"
25 #include "event.h"
```

```
26 #include "sphere.h"
27 #include "image.h"
28 #include "error.h"
29 #include "startstop.h"
30 #include "mouse.h"
31
32 #endif
```

2.8 image.h File Reference

```
#include "graphics.h"
```

Include dependency diagram for image.h

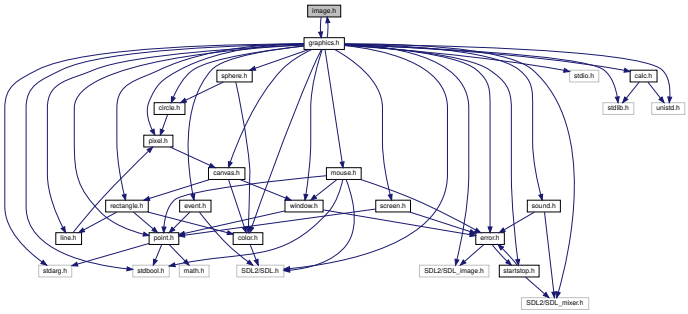

Figure 2.15: Dependency diagram

Included by dependency diagram for image.h

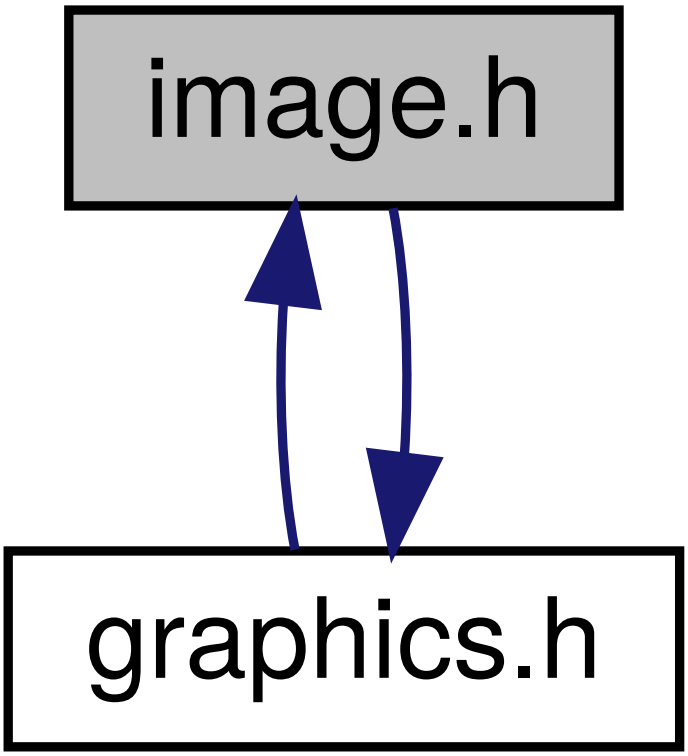

Figure 2.16: Dependency diagram

2.8.1 Data Structures

- struct Image

2.8.2 Functions

- void image_blit_naive (const Image * image)
- void image_blit_scaled (const Image * image)
- void image_load (Image * image, const char * pathToImg)
- void image_unload (Image * image)

2.8.3 Detailed Description

Definition in file image.h

```
1 #ifndef DEF_IMAGE_H
2 #define DEF_IMAGE_H
3
4 #include "graphics.h"
5
6 typedef struct {
7     SDL_Surface* surface;
8     Canvas* canvas;
9 } Image;
```

```
10
11 void image_blit_naive(const Image* image);
12
13 void image_blit_scaled(const Image* image);
14
15 void image_load(Image* image, const char* pathToImg);
16
17 void image_unload(Image* image);
18
19 #endif
```

2.9 line.h File Reference

```
#include "pixel.h"
```

Include dependency diagram for line.h

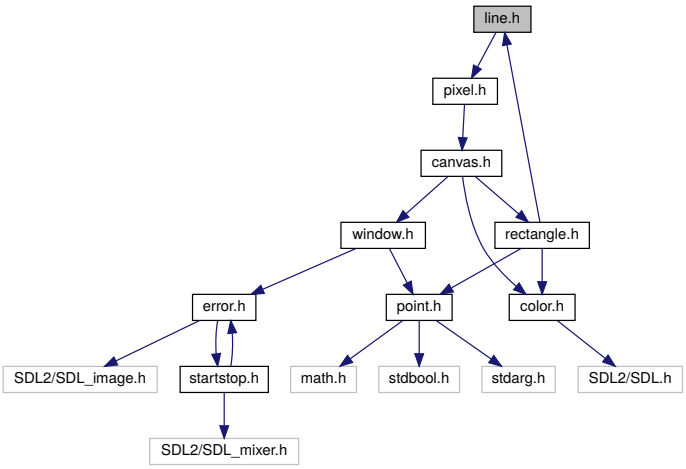

Figure 2.17: Dependency diagram

Included by dependency diagram for line.h

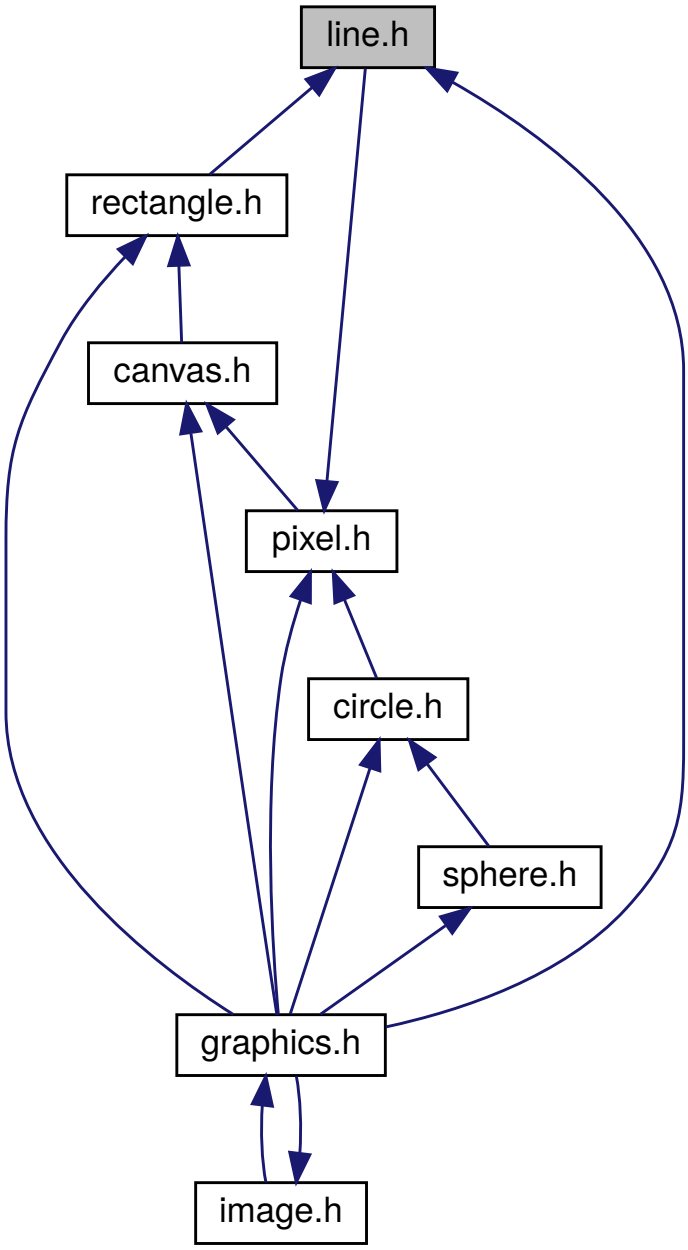

Figure 2.18: Dependency diagram

2.9.1 Data Structures

- struct Line

2.9.2 Functions

- void line_draw (const Line * line, const Color * color)
- void line_draw_bis (const Line * line, const Color * color)
- void line_draw_ter (const Line * line, const Color * color)

2.9.3 Detailed Description

Definition in file line.h

```
1 #ifndef DEF_LINE_H
2 #define DEF_LINE_H
3
4 #include "pixel.h"
5
6 typedef struct {
7     Point a;
8     Point b;
9     Canvas* canvas;
10 } Line;
11
12 void line_draw(const Line* line, const Color* color);
13
14 void line_draw_bis(const Line* line, const Color* color);
15
16 void line_draw_ter(const Line* line, const Color* color);
17
18 #endif
```

2.10 mouse.h File Reference

```
#include <stdbool.h>
```

```
#include <SDL2/SDL.h>
```

```
#include "error.h"
```

```
#include "point.h"
```

```
#include "window.h"
```

Include dependency diagram for mouse.h

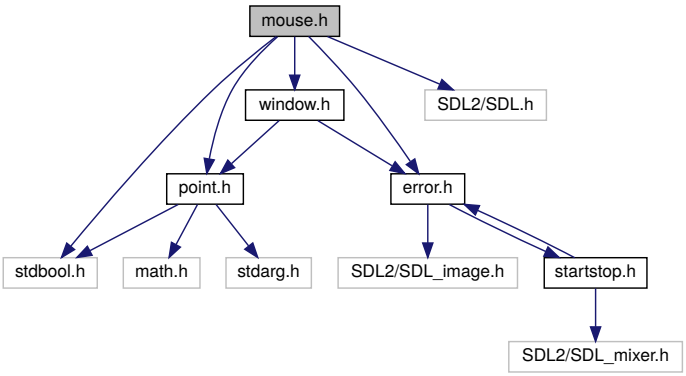

Figure 2.19: Dependency diagram

Included by dependency diagram for mouse.h

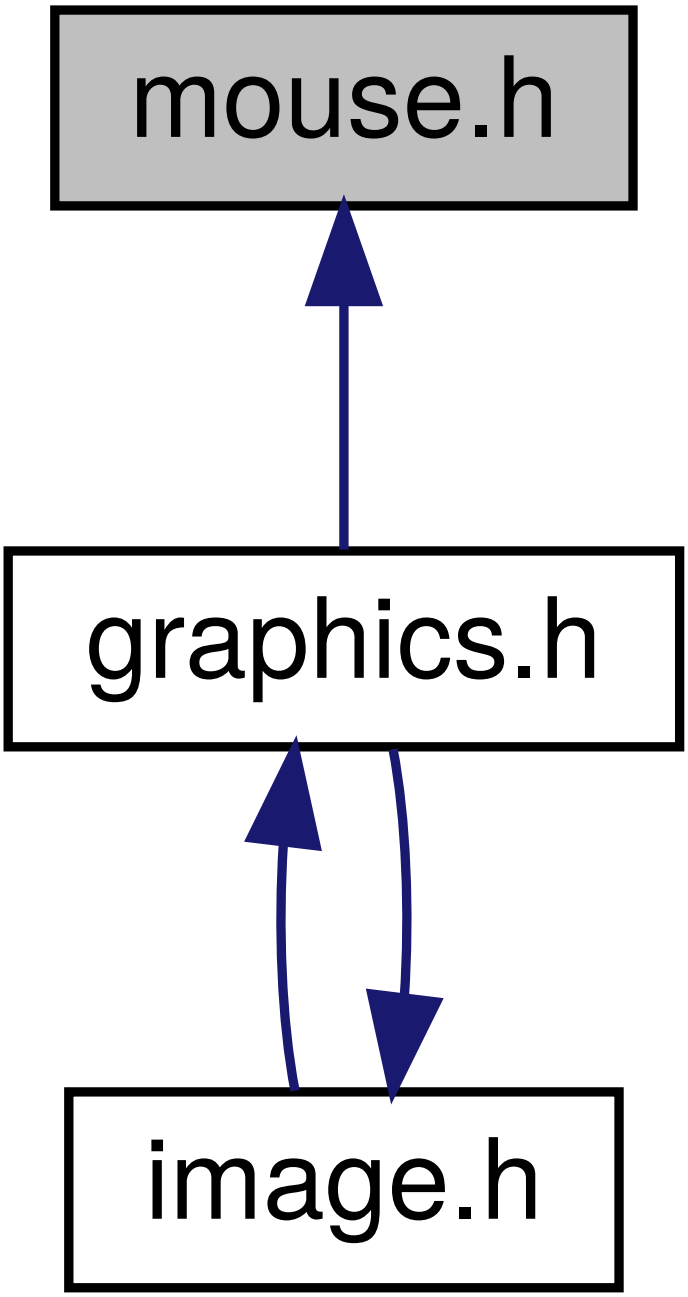

Figure 2.20: Dependency diagram

2.10.1 Functions

- void mouse_hide (void)
- void mouse_show (void)
- void mouse_wait_click (const Window * window, Point * click)
- bool mouse_is_hidden (void)
- bool mouse_is_shown (void)

2.10.2 Detailed Description

Definition in file mouse.h

```
1 #ifndef DEF_MOUSE_H
2 #define DEF_MOUSE_H
3
4 #include <stdbool.h>
5 #include <SDL2/SDL.h>
6 #include "error.h"
7 #include "point.h"
8 #include "window.h"
9
10 void mouse_hide(void);
11
12 void mouse_show(void);
13
14 void mouse_wait_click(const Window* window, Point* click);
15
16 bool mouse_is_hidden(void);
17
18 bool mouse_is_shown(void);
19
20 #endif
```

2.11 pixel.h File Reference

```
#include "canvas.h"
```

Include dependency diagram for pixel.h

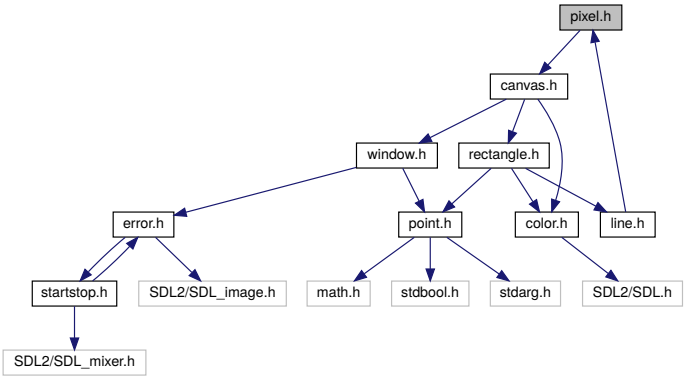

Figure 2.21: Dependency diagram

Included by dependency diagram for pixel.h

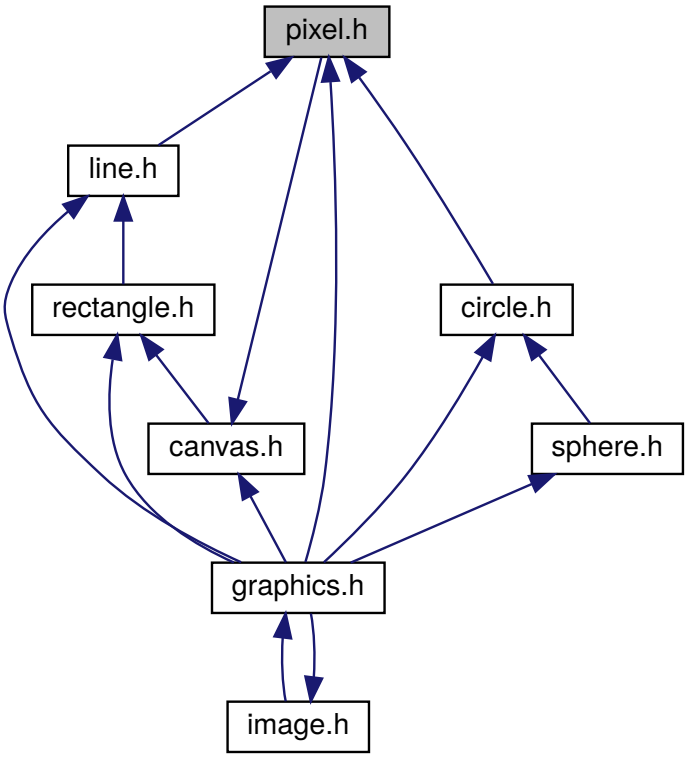

Figure 2.22: Dependency diagram

2.11.1 Data Structures

- struct Pixel

2.11.2 Functions

- void pixel_draw (const Pixel * pixel, const Color * color)

2.11.3 Detailed Description

Definition in file pixel.h

```
1 #ifndef DEF_PIXEL_H
2 #define DEF_PIXEL_H
3
4 #include "canvas.h"
5
6 typedef struct {
7     Point position;
8     Canvas* canvas;
9 } Pixel;
10
11 void pixel_draw(const Pixel* pixel, const Color* color);
12
13 #endif
```

2.12 point.h File Reference

```
#include <math.h>
```

```
#include <stdbool.h>
```

```
#include <stdarg.h>
```

Include dependency diagram for point.h

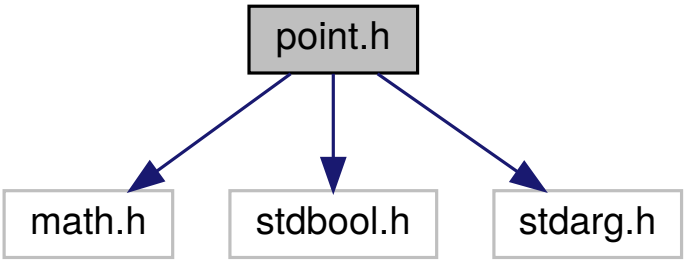

Figure 2.23: Dependency diagram

Included by dependency diagram for point.h

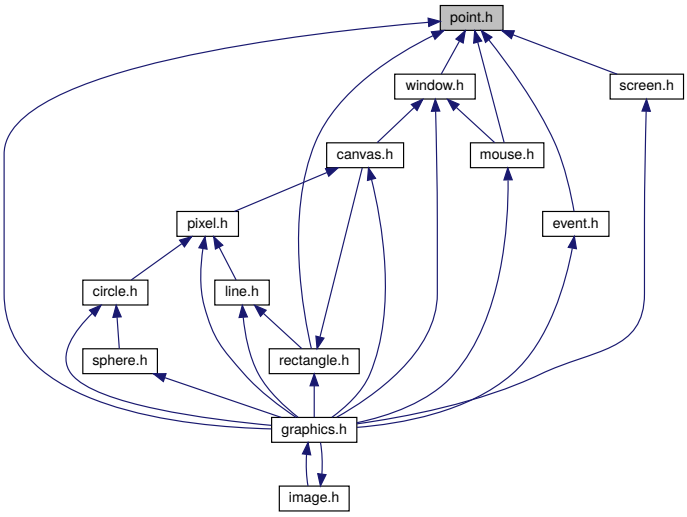

Figure 2.24: Dependency diagram

2.12.1 Data Structures

- struct Point

2.12.2 Functions

- bool point_are_equals (const Point p1, const Point p2)
- int point_distance (const Point a, const Point b)

- Point point_max_x (const Point a, const Point b)
- Point point_max_y (const Point a, const Point b)
- Point point_min_x (const Point a, const Point b)
- Point point_min_y (const Point a, const Point b)

2.12.3 Detailed Description

Definition in file point.h

```
1 #ifndef DEF_POINT_H
2 #define DEF_POINT_H
3
4 #include <math.h>
5 #include <stdbool.h>
6 #include <stdarg.h>
7
8 typedef struct {
9     int x;
10     int y;
11 } Point;
12
13 bool point_are_equals(const Point p1, const Point p2) __attribute__((const));
14
15 int point_distance(const Point a, const Point b);
16
17 Point point_max_x(const Point a, const Point b);
18
19 Point point_max_y(const Point a, const Point b);
20
21 Point point_min_x(const Point a, const Point b);
22
23 Point point_min_y(const Point a, const Point b);
24
25 #endif
```

2.13 rectangle.h File Reference

```
#include "point.h"
```

```
#include "line.h"
```

```
#include "color.h"
```

Include dependency diagram for rectangle.h

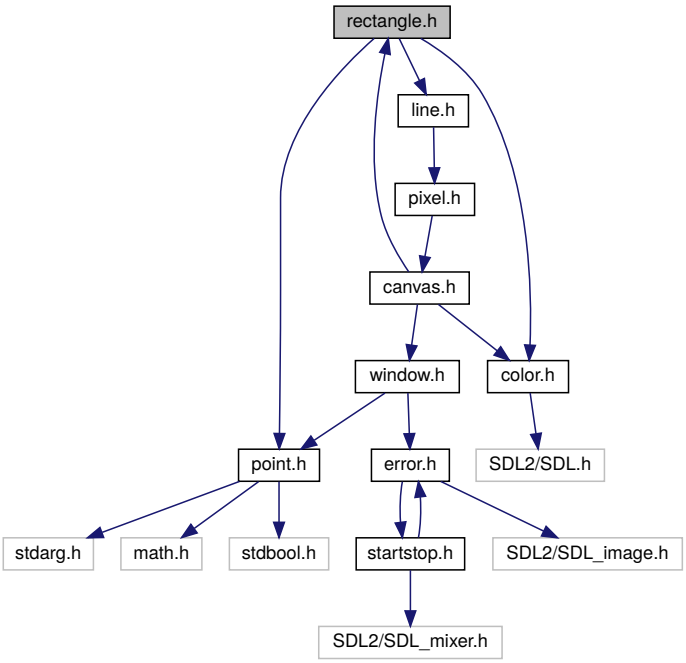

Figure 2.25: Dependency diagram

Included by dependency diagram for rectangle.h

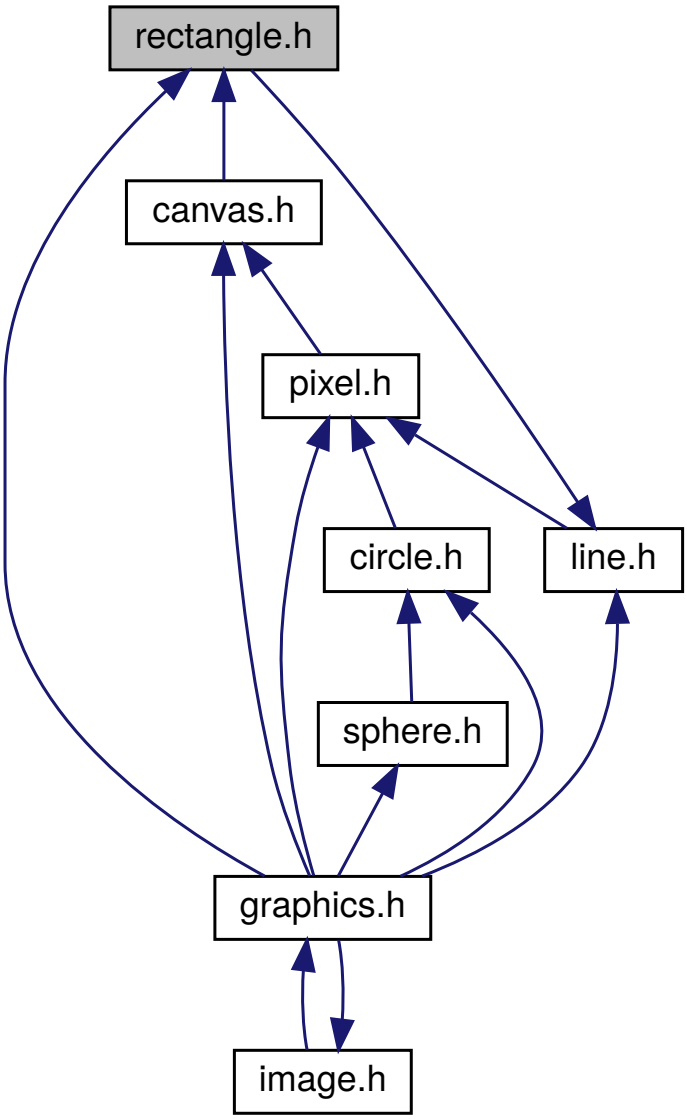

Figure 2.26: Dependency diagram

2.13.1 Data Structures

- struct Rectangle

2.13.2 Functions

- void rectangle_draw (const Rectangle * rectangle, const Color * color)
- void rectangle_draw_fill (const Rectangle * rectangle, const Color * color)
- bool rectangle_contains_point (const Rectangle * rect, const Point * p)
- bool rectangle_contains_absolute_point (const Rectangle * rect, const Point * p)

2.13.3 Detailed Description

Definition in file rectangle.h

```
1 #ifndef DEF_RECTANGLE_H
2 #define DEF_RECTANGLE_H
3
4 #include "point.h"
5 #include "line.h"
6 #include "color.h"
7
8 typedef struct {
9     Point origin;
10     Point size;
11     Canvas* canvas;
12 } Rectangle;
13
14 void rectangle_draw(const Rectangle* rectangle, const Color* color);
15
16 void rectangle_draw_fill(const Rectangle* rectangle, const Color* color);
17
18 bool rectangle_contains_point(const Rectangle* rect, const Point* p)  ↩
   __attribute__((pure));
19
20 bool rectangle_contains_absolute_point(const Rectangle* rect, const Point* p);
21
22 #endif
```

2.14 screen.h File Reference

```
#include "error.h"
```

```
#include "point.h"
```

Include dependency diagram for screen.h

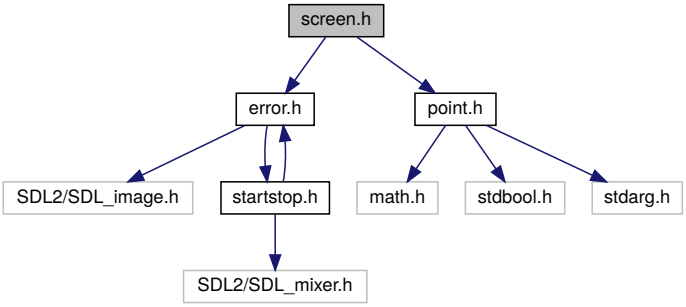

Figure 2.27: Dependency diagram

Included by dependency diagram for screen.h

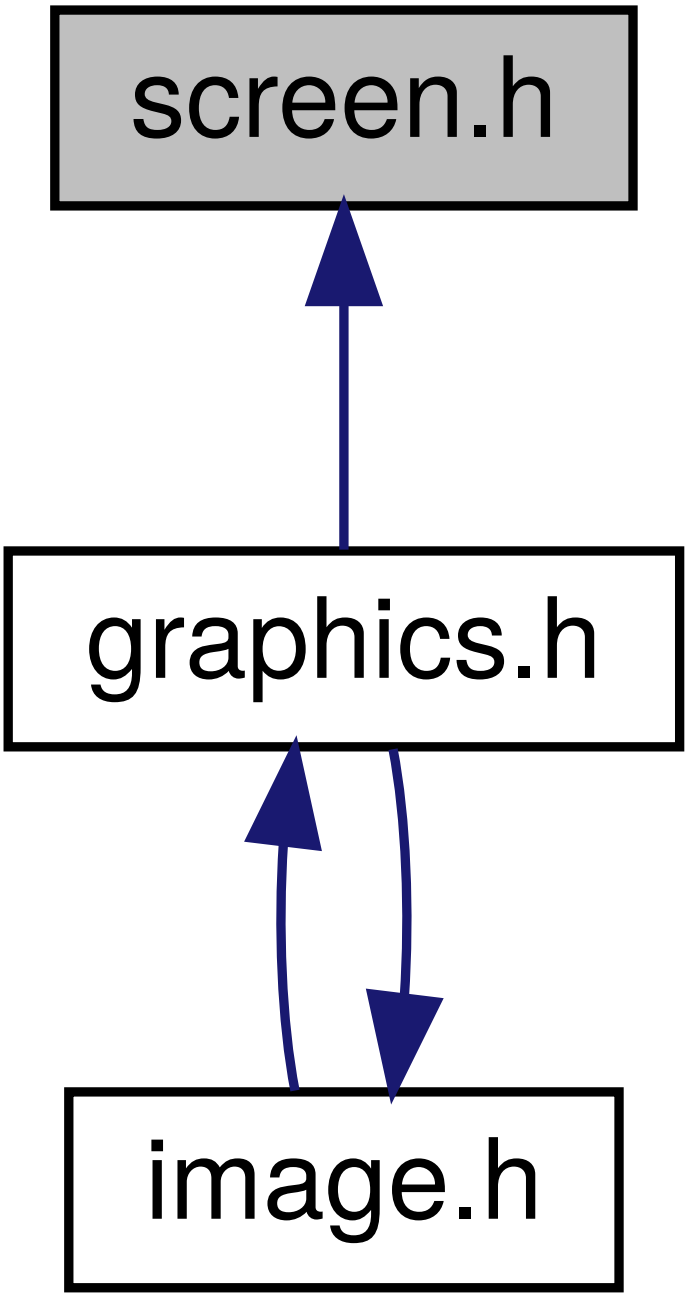

Figure 2.28: Dependency diagram

2.14.1 Functions

- void screen_get_size (Point * screenSize)

2.14.2 Detailed Description

Definition in file screen.h

```
1 #ifndef DEF_SCREEN_H
2 #define DEF_SCREEN_H
3
```

```
4 #include "error.h"
5 #include "point.h"
6
7 void screen_get_size(Point* screenSize);
8
9 #endif
```

2.15 sound.h File Reference

```
#include <SDL2/SDL_mixer.h>
```

```
#include "error.h"
```

Include dependency diagram for sound.h

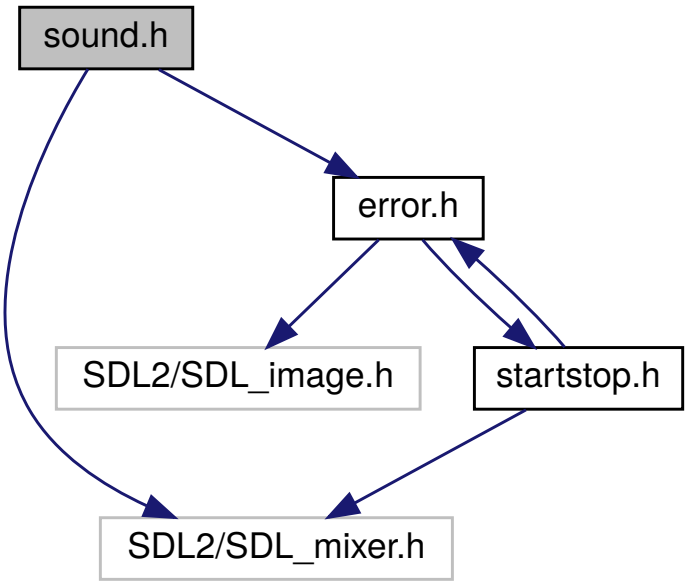

Figure 2.29: Dependency diagram

Included by dependency diagram for sound.h

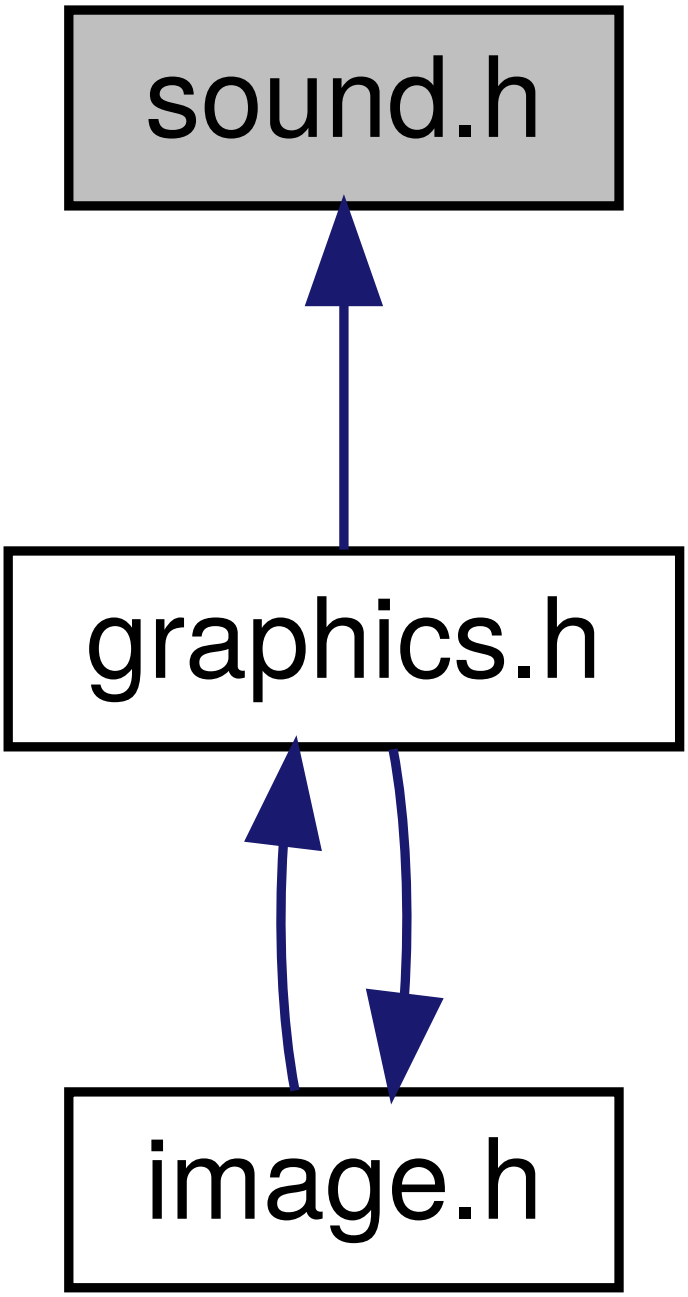

Figure 2.30: Dependency diagram

2.15.1 Data Structures

- struct Sound

2.15.2 Functions

- void sound_load (const char * fileName, Sound * sound)
- void sound_play (const Sound * music)
- void sound_play_once (const Sound * music)

- void sound_free (Sound * sound)
- void sound_stop (void)
- void sound_pause (void)
- void sound_resume (void)

2.15.3 Detailed Description

Definition in file sound.h

```
1 #ifndef DEF_SOUND_H
2 #define DEF_SOUND_H
3
4 #include <SDL2/SDL_mixer.h>
5 #include "error.h"
6
7 typedef struct {
8     Mix_Music* content;
9 } Sound;
10
11 void sound_load(const char* fileName, Sound* sound);
12
13 void sound_play(const Sound* music);
14
15 void sound_play_once(const Sound* music);
16
17 void sound_free(Sound* sound);
18
19 void sound_stop(void);
20
21 void sound_pause(void);
22
23 void sound_resume(void);
24
25 #endif
```

2.16 sphere.h File Reference

```
#include "circle.h"
```

```
#include "color.h"
```

Include dependency diagram for sphere.h

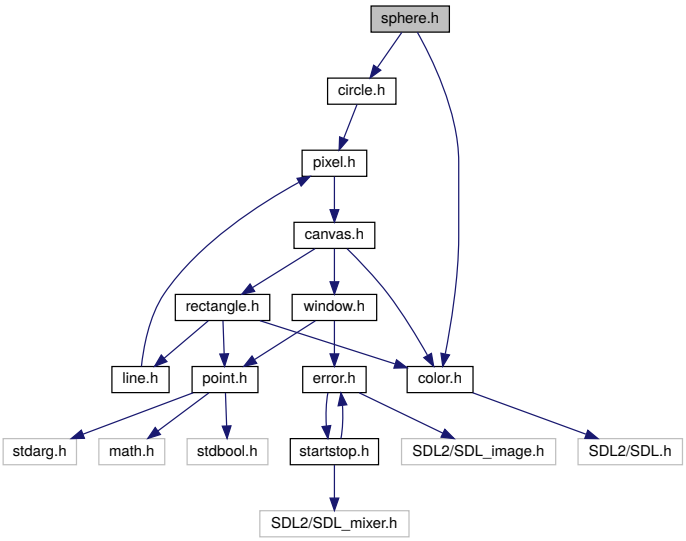

Figure 2.31: Dependency diagram

Included by dependency diagram for sphere.h

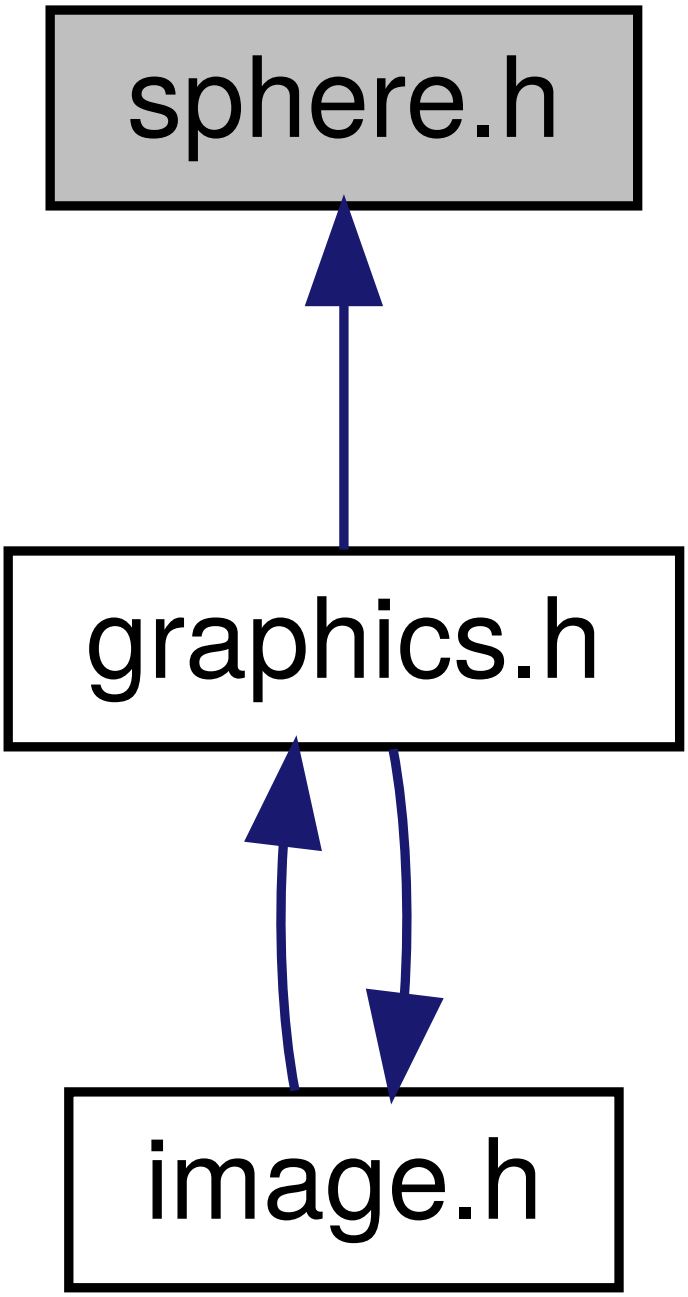

Figure 2.32: Dependency diagram

2.16.1 Data Structures

- struct Sphere

2.16.2 Functions

- void sphere_draw_fill (const Sphere * sphere, const Color * color)

2.16.3 Detailed Description

Definition in file sphere.h

```
1 #ifndef DEF_SPHERE_H
2 #define DEF_SPHERE_H
3
4 #include "circle.h"
5 #include "color.h"
6
7 #pragma pack(push, 1)
8 typedef struct {
9     Point center;
10     int radius;
11     Canvas* canvas;
12 } Sphere;
13 #pragma pack(pop)
14
15 void sphere_draw_fill(const Sphere* sphere, const Color* color);
16
17 #endif
```

2.17 startstop.h File Reference

```
#include <SDL2/SDL_mixer.h>
```

```
#include "error.h"
```

Include dependency diagram for startstop.h

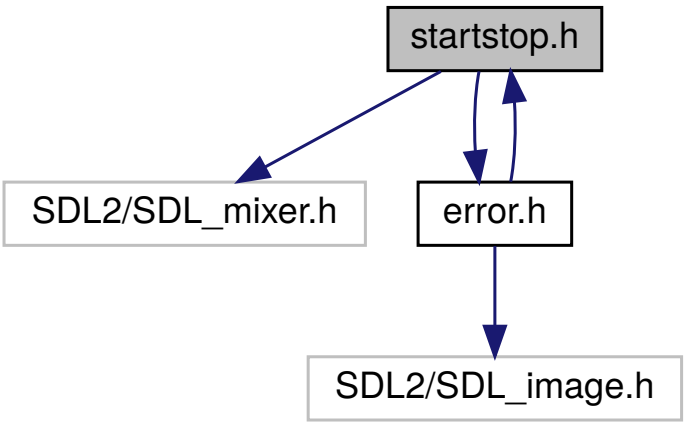

Figure 2.33: Dependency diagram

Included by dependency diagram for startstop.h

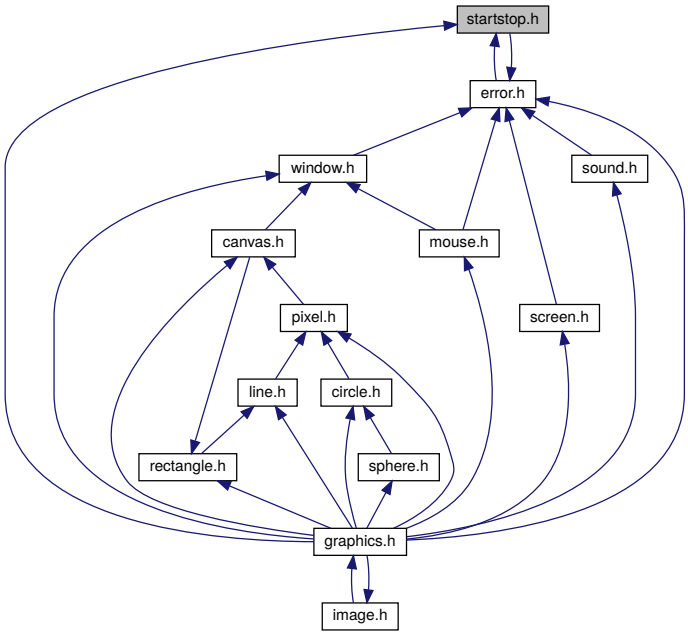

Figure 2.34: Dependency diagram

2.17.1 Functions

- void graphics_start (const Uint32 flags)
- void graphics_stop (void)

2.17.2 Detailed Description

Definition in file startstop.h

```
1 #ifndef DEF_STARTSTOP_H
2 #define DEF_STARTSTOP_H
3
4 #include <SDL2/SDL_mixer.h>
5 #include "error.h"
6
7 void graphics_start(const Uint32 flags);
8
9 void graphics_stop(void);
10
11 #endif
```

2.18 window.h File Reference

```
#include "error.h"
```

```
#include "point.h"
```

Include dependency diagram for window.h

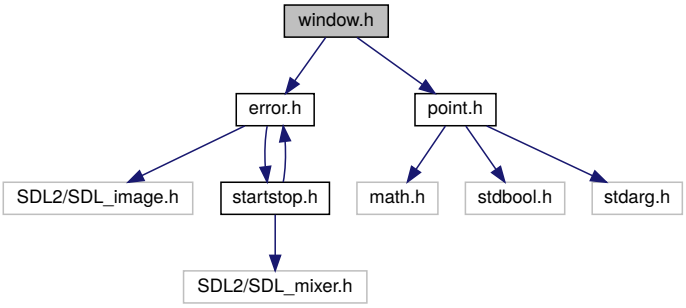

Figure 2.35: Dependency diagram

Included by dependency diagram for window.h

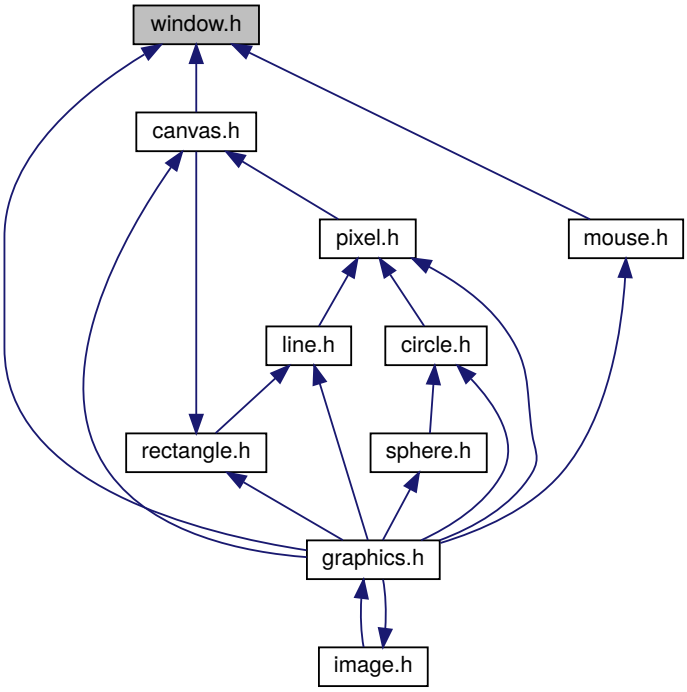

Figure 2.36: Dependency diagram

2.18.1 Data Structures

- struct Window

2.18.2 Functions

- void window_create (Window * window, char * title, const Point * position, const Point * size, const Uint32 flags)
- void window_destroy (Window * window)
- void window_update (Window * window)

2.18.3 Detailed Description

Definition in file window.h

```
1 #ifndef DEF_WINDOW_H
2 #define DEF_WINDOW_H
3
4 #include "error.h"
5 #include "point.h"
6
7 typedef struct {
8     SDL_Window* window;
9     char* title;
10     Point position;
11     Point size;
12 } Window;
13
14 void window_create(Window* window, char* title, const Point* position, const ↩
   Point* size, const Uint32 flags);
15
16 void window_destroy(Window* window);
17
18 void window_update(Window* window);
19
20 #endif
```

Chapter 3

Directory Documentation

3.1 head Directory Reference

Directory dependency diagram for head

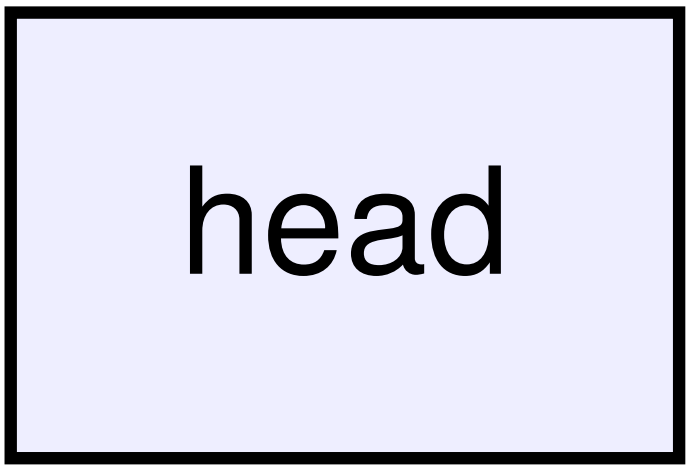

Figure 3.1: Directory Dependency diagram

3.1.1 File

- file calc.h
- file canvas.h
- file circle.h
- file color.h
- file error.h
- file event.h
- file graphics.h

- file image.h
- file line.h
- file mouse.h
- file pixel.h
- file point.h
- file rectangle.h
- file screen.h
- file sound.h
- file sphere.h
- file startstop.h
- file window.h

3.1.2 Detailed Description

Directory location is /home/leo/Programmation/C/graphics/head/