ProjetTechnoL3TP1 0.0.1

Generated by Doxygen 1.8.11

# **Contents**

1	Clas	es Index	1
	1.1	Class List	1
2	File	Index	3
	2.1	File List	3
3	Clas	es Documentation	5
	3.1	ei_app_event_t Struct Reference	5
		3.1.1 Detailed Description	5
	3.2	ei_color_t Struct Reference	5
		3.2.1 Detailed Description	6
		3.2.2 Member Data Documentation	6
		3.2.2.1 alpha	6
	3.3	ei_display_event_t Struct Reference	6
		3.3.1 Detailed Description	6
	3.4	ei_event_t Struct Reference	7
		3.4.1 Detailed Description	7
	3.5	ei_key_event_t Struct Reference	7
		3.5.1 Detailed Description	8
	3.6	ei_linked_point_t Struct Reference	8
		3.6.1 Detailed Description	9
	3.7	ei_linked_rect_t Struct Reference	9
		3.7.1 Detailed Description	9
	3.8	ei linked tag t Struct Reference	10

iv CONTENTS

		3.8.1	Detailed Description	10
	3.9	ei_mou	use_event_t Struct Reference	10
		3.9.1	Detailed Description	11
		3.9.2	Member Data Documentation	11
			3.9.2.1 button_number	11
	3.10	ei_poir	t_t Struct Reference	11
		3.10.1	Detailed Description	11
		3.10.2	Member Data Documentation	11
			3.10.2.1 y	11
	3.11	ei_rect	_t Struct Reference	12
		3.11.1	Detailed Description	12
	3.12	ei_size	_t Struct Reference	12
		3.12.1	Detailed Description	13
	3.13	ei_touc	ch_event_t Struct Reference	13
		3.13.1	Detailed Description	13
		3.13.2	Member Data Documentation	14
			3.13.2.1 primary	14
			3.13.2.2 touch_id	14
4	File I	Docume	entation	15
	4.1	include	/ei_event.h File Reference	15
		4.1.1	Detailed Description	16
		4.1.2	Enumeration Type Documentation	16
			4.1.2.1 ei_eventtype_t	16
			4.1.2.2 ei_modifier_key_t	17
	4.2	include	/ei_main.h File Reference	17
		4.2.1	Detailed Description	17
		4.2.2	Function Documentation	17
			4.2.2.1 ei_main(int argc, char *argv[])	17
	4.3	include	/ei_types.h File Reference	18
		4.3.1	Detailed Description	19

CONTENTS

	4.3.2	Typedef I	Documentation	20
		4.3.2.1	ei_font_t	20
	4.3.3	Enumera	tion Type Documentation	20
		4.3.3.1	ei_anchor_t	20
		4.3.3.2	ei_axis_set_t	20
		4.3.3.3	ei_relief_t	20
4.4	include	/hw_interf	ace.h File Reference	21
	4.4.1	Detailed	Description	22
	4.4.2	Function	Documentation	22
		4.4.2.1	hw_create_window(ei_size_t *size, const ei_bool_t fullScreen)	22
		4.4.2.2	hw_event_post_app(void *user_param)	23
		4.4.2.3	hw_event_wait_next(struct ei_event_t *event)	23
		4.4.2.4	hw_get_pixel(const ei_surface_t surface, const ei_point_t pos)	23
		4.4.2.5	hw_image_load(const char *filename)	23
		4.4.2.6	hw_now()	24
		4.4.2.7	hw_put_pixel(const ei_surface_t surface, const ei_point_t pos, const ei_color_t color)	24
		4.4.2.8	hw_surface_create(const ei_surface_t root, const ei_size_t *size)	24
		4.4.2.9	hw_surface_free(ei_surface_t surface)	25
		4.4.2.10	hw_surface_get_rect(const ei_surface_t surface)	25
		4.4.2.11	hw_surface_get_size(const ei_surface_t surface)	25
		4.4.2.12	hw_surface_lock(ei_surface_t surface)	25
		4.4.2.13	hw_surface_unlock(ei_surface_t surface)	26
		4.4.2.14	hw_surface_update_rects(const ei_linked_rect_t *rects)	26
		4.4.2.15	hw_text_compute_size(const char *text, const ei_font_t font, int *width, int *height)	26
		4.4.2.16	hw_text_create_surface(const char *text, const ei_font_t font, const ei_color_ ← t *color)	26
		4.4.2.17	hw_text_font_create(const char *filename, int size)	27
		4.4.2.18	hw_text_font_free(ei_font_t font)	28
		4.4.2.19	hw_wait(int s_delay)	28

29

Index

# **Chapter 1**

# **Class Index**

# 1.1 Class List

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

ei_app_event_t	
The event parameter for application defined event types	5
ei_color_t	
A color with transparency	5
ei_display_event_t	
The event parameter for display-related event types	6
ei_event_t	
Describes an event	7
ei_key_event_t	
The event parameter for keyboard-related event types	7
ei_linked_point_t	
A point plus a pointer to create a linked list	8
ei_linked_rect_t	
A rectangle plus a pointer to create a linked list	9
ei_linked_tag_t	
0 1	10
ei_mouse_event_t	
71	10
ei_point_t	
F	11
ei_rect_t	
	12
ei_size_t	
3	12
ei_touch_event_t	
The event parameter for mouse-related event types	13

2 Class Index

# Chapter 2

# File Index

# 2.1 File List

Here is a list of all documented files with brief descriptions:

include/ei_event.h	
Allows the binding and unbinding of callbacks to events	15
include/ei_main.h	
Declares the "ei_main" function: the main function of programs built with the libei	17
include/ei_types.h	
Type, constant, and global definitions for the ei library	18
include/hw_interface.h	
Low level interface with the graphic hadware. This interface is based on the SDL library	21

File Index

# **Chapter 3**

# **Class Documentation**

# 3.1 ei\_app\_event\_t Struct Reference

The event parameter for application defined event types.

```
#include <ei_event.h>
```

## **Public Attributes**

void \* user\_param

# 3.1.1 Detailed Description

The event parameter for application defined event types.

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

• include/ei\_event.h

# 3.2 ei\_color\_t Struct Reference

A color with transparency.

```
#include <ei_types.h>
```

# **Public Attributes**

• unsigned char red

The red component of the color.

• unsigned char green

The green component of the color.

· unsigned char blue

The blue component of the color.

• unsigned char alpha

6 Class Documentation

# 3.2.1 Detailed Description

A color with transparency.

Each channel is represented as an 8 bits unsigned interger, hence channel's minimum value is 0, maximum is 255.

## 3.2.2 Member Data Documentation

```
3.2.2.1 unsigned char ei_color_t::alpha
```

The transparency of the color. 0 is invisible,

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

• include/ei\_types.h

# 3.3 ei\_display\_event\_t Struct Reference

The event parameter for display-related event types.

```
#include <ei_event.h>
```

#### **Public Attributes**

· ei\_bool\_t resized

The window has been resized.

· ei bool t closed

The close button of the window has been pressed.

ei\_bool\_t switched\_out

The window is no longer active.

ei\_bool\_t switched\_in

The window is active once again.

# 3.3.1 Detailed Description

The event parameter for display-related event types.

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

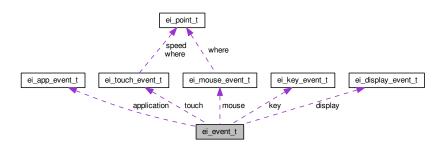
• include/ei\_event.h

# 3.4 ei\_event\_t Struct Reference

Describes an event.

```
#include <ei_event.h>
```

Collaboration diagram for ei\_event\_t:



## **Public Attributes**

Event parameters for touch-related events (see ei\_touch\_event\_t). ei\_app\_event\_t application

Event parameters for application-related events (see ei\_app\_event\_t). } param

## 3.4.1 Detailed Description

Describes an event.

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

• include/ei\_event.h

# 3.5 ei\_key\_event\_t Struct Reference

The event parameter for keyboard-related event types.

```
#include <ei_event.h>
```

8 Class Documentation

# **Public Attributes**

• int key\_sym

The keyboard key symbol (see allegro5/keycodes.h).

· int unichar

For ei\_ev\_keychar, a Unicode code point (character).

• ei\_modifier\_mask\_t modifier\_mask

The state of the modifier keys at the time of the event.

# 3.5.1 Detailed Description

The event parameter for keyboard-related event types.

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

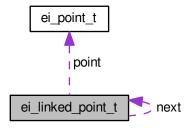
• include/ei\_event.h

# 3.6 ei\_linked\_point\_t Struct Reference

A point plus a pointer to create a linked list.

```
#include <ei_types.h>
```

Collaboration diagram for ei\_linked\_point\_t:



# **Public Attributes**

ei\_point\_t point

The point.

struct ei\_linked\_point\_t \* next

The pointer to the next element in the linked list.

# 3.6.1 Detailed Description

A point plus a pointer to create a linked list.

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

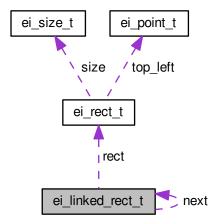
• include/ei\_types.h

# 3.7 ei\_linked\_rect\_t Struct Reference

A rectangle plus a pointer to create a linked list.

```
#include <ei_types.h>
```

Collaboration diagram for ei\_linked\_rect\_t:



## **Public Attributes**

• ei\_rect\_t rect

The rectangle.

struct ei\_linked\_rect\_t \* next

The pointer to the next element in the linked list.

# 3.7.1 Detailed Description

A rectangle plus a pointer to create a linked list.

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

• include/ei\_types.h

10 Class Documentation

# 3.8 ei\_linked\_tag\_t Struct Reference

A tag and a pointer to create a linked list.

```
#include <ei_event.h>
```

Collaboration diagram for ei\_linked\_tag\_t:



## **Public Attributes**

- ei\_tag\_t tag
- struct ei\_linked\_tag\_t \* next

# 3.8.1 Detailed Description

A tag and a pointer to create a linked list.

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

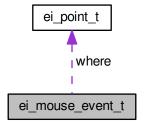
• include/ei\_event.h

# 3.9 ei\_mouse\_event\_t Struct Reference

The event parameter for mouse-related event types.

```
#include <ei_event.h>
```

Collaboration diagram for ei\_mouse\_event\_t:



## **Public Attributes**

ei\_point\_t where

Where the mouse pointer was at the time of the event.

· int button\_number

# 3.9.1 Detailed Description

The event parameter for mouse-related event types.

#### 3.9.2 Member Data Documentation

3.9.2.1 int ei\_mouse\_event\_t::button\_number

The number of the button that was pressed or released.

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

• include/ei\_event.h

# 3.10 ei\_point\_t Struct Reference

A 2-D point with integer coordinates.

```
#include <ei_types.h>
```

## **Public Attributes**

int x

The abscissa of the point. The origin is on the left side of the image.

• int y

# 3.10.1 Detailed Description

A 2-D point with integer coordinates.

## 3.10.2 Member Data Documentation

3.10.2.1 int ei\_point\_t::y

The ordinate of the point, the origin is at the top of the image,

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

• include/ei\_types.h

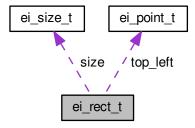
12 Class Documentation

# 3.11 ei\_rect\_t Struct Reference

A rectangle defined by its top-left corner, and its size.

```
#include <ei_types.h>
```

Collaboration diagram for ei\_rect\_t:



# **Public Attributes**

• ei\_point\_t top\_left

Coordinates of the top-left corner of the rectangle.

• ei\_size\_t size

Size of the rectangle.

# 3.11.1 Detailed Description

A rectangle defined by its top-left corner, and its size.

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

• include/ei\_types.h

# 3.12 ei\_size\_t Struct Reference

A 2-D size with integer dimensions.

```
#include <ei_types.h>
```

## **Public Attributes**

- int width
- int height

# 3.12.1 Detailed Description

A 2-D size with integer dimensions.

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

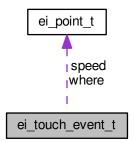
• include/ei\_types.h

# 3.13 ei\_touch\_event\_t Struct Reference

The event parameter for mouse-related event types.

```
#include <ei_event.h>
```

Collaboration diagram for ei\_touch\_event\_t:



# **Public Attributes**

· ei\_point\_t where

Where the touch was at the time of the event.

• ei\_point\_t speed

Movement speed in pixels.

- int touch\_id
- ei\_bool\_t primary

it will stay the same for events from the same finger until the touch ends.

# 3.13.1 Detailed Description

The event parameter for mouse-related event types.

14 Class Documentation

# 3.13.2 Member Data Documentation

3.13.2.1 ei\_bool\_t ei\_touch\_event\_t::primary

it will stay the same for events from the same finger until the touch ends.

Whether this is the only/first touch or an additional touch.

3.13.2.2 int ei\_touch\_event\_t::touch\_id

An identifier for this touch. If supported by the device

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

• include/ei\_event.h

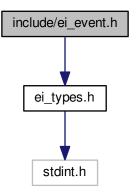
# **Chapter 4**

# **File Documentation**

# 4.1 include/ei\_event.h File Reference

Allows the binding and unbinding of callbacks to events.

```
#include "ei_types.h"
Include dependency graph for ei_event.h:
```



# Classes

• struct ei\_linked\_tag\_t

A tag and a pointer to create a linked list.

• struct ei\_display\_event\_t

The event parameter for display-related event types.

struct ei\_key\_event\_t

The event parameter for keyboard-related event types.

• struct ei\_mouse\_event\_t

The event parameter for mouse-related event types.

```
• struct ei_touch_event_t
```

The event parameter for mouse-related event types.

• struct ei\_app\_event\_t

The event parameter for application defined event types.

• struct ei\_event\_t

Describes an event.

# **Typedefs**

• typedef char \* ei tag t

A string that can be attached to a widget. All widget have the tag of the name of their widget class, and the tag "all".

typedef struct ei\_linked\_tag\_t ei\_linked\_tag\_t

A tag and a pointer to create a linked list.

• typedef uint32 t ei modifier mask t

A bitfield indicating which of the modifier keys are currently pressed.

typedef struct ei\_event\_t ei\_event\_t

Describes an event.

#### **Enumerations**

```
enum ei_eventtype_t {
    ei_ev_none = 0, ei_ev_app, ei_ev_display, ei_ev_keydown,
    ei_ev_keyup, ei_ev_keychar, ei_ev_mouse_buttondown, ei_ev_mouse_buttonup,
    ei_ev_mouse_move, ei_ev_touch_begin, ei_ev_touch_end, ei_ev_touch_move,
    ei_ev_last }
    The types of events.
enum ei_modifier_key_t {
    ei_mod_shift = 0x00001, ei_mod_ctrl = 0x00002, ei_mod_alt = 0x00004, ei_mod_meta_left = 0x00008,
    ei_mod_meta_right = 0x00010, ei_mod_alt_grad = 0x000040 }
    The modifier keys (shift, alt, etc.)
```

#### 4.1.1 Detailed Description

Allows the binding and unbinding of callbacks to events.

Author

Created by François Bérard on 30.12.11. Copyright 2011 Ensimag. All rights reserved.

# 4.1.2 Enumeration Type Documentation

```
4.1.2.1 enum ei eventtype t
```

The types of events.

#### **Enumerator**

ei\_ev\_none No event, used on an un-initialized structure.

```
ei_ev_app An application event, created by hw_event_post_app.
ei_ev_display A display / window event.
ei_ev_keydown A keyboard key has been pressed.
ei_ev_keyup A keyboard key has been released.
ei_ev_keychar A character was typed on the keyboard.
ei_ev_mouse_buttondown A mouse button has been pressed.
ei_ev_mouse_buttonup A mouse button has been released.
ei_ev_mouse_move The mouse has moved.
ei_ev_touch_begin The touch input device registered a new touch.
ei_ev_touch_end A touch ended.
ei_ev_touch_move The position of a touch changed.
ei_ev_last Last event type, its value is the number of event types.
```

#### 4.1.2.2 enum ei\_modifier\_key\_t

The modifier keys (shift, alt, etc.)

#### **Enumerator**

```
ei_mod_shift The "shift" key.
ei_mod_ctrl The "control" key.
ei_mod_alt The "alternate" key at the left of the space bar.
ei_mod_meta_left The "meta" (command) key at the left of the space bar.
ei_mod_meta_right The "meta" (command) key at the right of the space bar.
ei_mod_alt_grad The "alternate" key at the right of the space bar.
```

# 4.2 include/ei\_main.h File Reference

Declares the "ei main" function: the main function of programs built with the libei.

#### **Functions**

```
    int ei_main (int argc, char *argv[])
    The main function of the program.
```

## 4.2.1 Detailed Description

Declares the "ei\_main" function: the main function of programs built with the libei.

#### **Author**

Created by François Bérard on 30.12.11. Copyright 2011 Ensimag. All rights reserved.

### 4.2.2 Function Documentation

```
4.2.2.1 int ei_main ( int argc, char * argv[] )
```

The main function of the program.

Programmers must not define their main function in a function called "main", because the "main" function is defined by SDL and linked with in the libeibase library.

## **Parameters**

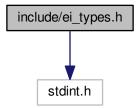
#### Returns

An error code: 0 means ok, 1 means error.

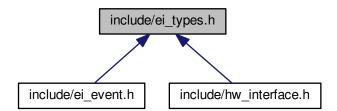
# 4.3 include/ei\_types.h File Reference

Type, constant, and global definitions for the ei library.

#include <stdint.h>
Include dependency graph for ei\_types.h:



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



# Classes

- struct ei\_point\_t
  - A 2-D point with integer coordinates.
- struct ei\_size\_t

A 2-D size with integer dimensions.

• struct ei\_rect\_t

A rectangle defined by its top-left corner, and its size.

struct ei\_linked\_rect\_t

A rectangle plus a pointer to create a linked list.

• struct ei\_linked\_point\_t

A point plus a pointer to create a linked list.

• struct ei\_color\_t

A color with transparency.

## **Typedefs**

• typedef struct ei\_linked\_rect\_t ei\_linked\_rect\_t

A rectangle plus a pointer to create a linked list.

typedef struct ei\_linked\_point\_t ei\_linked\_point\_t

A point plus a pointer to create a linked list.

typedef void \* ei\_font\_t

An opaque type for handling fonts.

#### **Enumerations**

```
    enum ei_bool_t { EI_FALSE = 0, EI_TRUE = 1 }
```

The boolean type used in the library.

```
enum ei_anchor_t {
```

```
ei_anc_none = 0, ei_anc_center, ei_anc_north, ei_anc_northeast,
ei_anc_east, ei_anc_southeast, ei_anc_south, ei_anc_southwest,
ei_anc_west, ei_anc_northwest }
```

Identifies one particular point of a rectangle.

• enum ei\_relief\_t { ei\_relief\_none = 0, ei\_relief\_raised, ei\_relief\_sunken }

Type of relief.

enum ei\_axis\_set\_t { ei\_axis\_none = 0, ei\_axis\_x, ei\_axis\_y, ei\_axis\_both }

Set of axis.

# **Variables**

• ei\_font\_t ei\_default\_font

The default font used in widgets.

## 4.3.1 Detailed Description

Type, constant, and global definitions for the ei library.

Created by François Bérard on 18.12.11. Copyright 2011 Ensimag. All rights reserved.

```
4.3.2 Typedef Documentation
```

```
4.3.2.1 typedef void* ei_font_t
```

An opaque type for handling fonts.

Fonts are created by calling hw text font create and released by calling hw text font free.

## 4.3.3 Enumeration Type Documentation

```
4.3.3.1 enum ei_anchor_t
```

Identifies one particular point of a rectangle.

#### **Enumerator**

```
ei_anc_none No anchor defined.
```

ei\_anc\_center Anchor in the center.

ei\_anc\_north Anchor on the top side, centered horizontally.

ei\_anc\_northeast Anchor on the top-right corner.

ei\_anc\_east Anchor on the right side, centered vertically.

ei\_anc\_southeast Anchor on the bottom-right corner.

ei\_anc\_south Anchor on the bottom side, centered horizontally.

ei\_anc\_southwest Anchor on the bottom-left corner.

ei\_anc\_west Anchor on the left side, centered vertically.

ei\_anc\_northwest Anchor on the top-left corner.

```
4.3.3.2 enum ei_axis_set_t
```

Set of axis.

#### Enumerator

```
ei_axis_none No axis.
```

ei\_axis\_x Horizontal axis.

ei\_axis\_y Vertical axis.

ei\_axis\_both Both horizontal and vertical axis.

```
4.3.3.3 enum ei_relief_t
```

Type of relief.

#### Enumerator

```
ei_relief_none No relief (i.e. flat).
```

ei\_relief\_raised Above the screen.

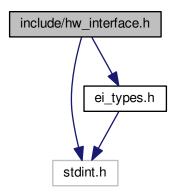
ei\_relief\_sunken Inside the screen.

# 4.4 include/hw\_interface.h File Reference

Low level interface with the graphic hadware. This interface is based on the SDL library.

```
#include <stdint.h>
#include "ei_types.h"
```

Include dependency graph for hw\_interface.h:



# **Typedefs**

• typedef void \* ei\_surface\_t

Surface hidden type. A surface represents a 2 dimentional array of pixels where drawing can be done. The displayed screen itself is represented by a surface, it is accessed by <a href="https://hw\_create\_window">hw\_create\_window</a>. Other "offscreen" surfaces can be created by <a href="https://hw\_surface\_create">hw\_surface\_create</a>.

# **Functions**

• void hw\_init ()

Initialises access to the low-level operating system services.

void hw\_quit ()

Closes the access to the low-level operating system services.

• ei\_surface\_t hw\_create\_window (ei\_size\_t \*size, const ei\_bool\_t fullScreen)

Opens the main graphical window of the application.

• ei\_surface\_t hw\_surface\_create (const ei\_surface\_t root, const ei\_size\_t \*size)

Allocates an off-screen drawing surface.

void hw\_surface\_free (ei\_surface\_t surface)

Frees a surface allocated by hw\_surface\_create. This must be called on an unlocked surface (see hw\_surface\_create. unlock).

· void hw surface lock (ei surface t surface)

Gains exclusive access to a surface. Every call to this function must be matched by a call to <a href="https://mww.surface\_unlock">hw\_surface\_unlock</a>. The address of the pixel buffer may change while the surface is unlocked. Thus, <a href="https://mws.urface\_get\_buffer must called after each call to this function">hw\_surface\_get\_buffer must called after each call to this function</a>.

void hw\_surface\_unlock (ei\_surface\_t surface)

Releases the exclusive access to a surface that was locked by hw surface lock.

ei\_size\_t hw\_surface\_get\_size (const ei\_surface\_t surface)

Returns the size of a surface.

void hw surface update rects (const ei linked rect t \*rects)

Requests that a list of rectangular regions of the root surface be updated on screen.

ei\_rect\_t hw\_surface\_get\_rect (const ei\_surface\_t surface)

Returns the rectangle of a surface (origin and size).

• ei color thw get pixel (const ei surface t surface, const ei point t pos)

Returns the color of the pixel at pos.

void hw\_put\_pixel (const ei\_surface\_t surface, const ei\_point\_t pos, const ei\_color\_t color)

Draw the pixel of the surface at pos.

• ei surface thw image load (const char \*filename)

Creates a surface and loads into it an image read from a file. The caller is responsible to release this surface (hw\_ \cup surface free) when it is no more needed.

• ei\_font\_t hw\_text\_font\_create (const char \*filename, int size)

Creates a font that can be used to render text. The font must be freed by calling hw\_text\_font\_free.

void hw text font free (ei font t font)

Frees a font created by hw text font create.

void hw text compute size (const char \*text, const ei font t font, int \*width, int \*height)

Computes the size of a text surface givent the font and the text.

ei\_surface\_t hw\_text\_create\_surface (const char \*text, const ei\_font\_t font, const ei\_color\_t \*color)

Creates a surface containing a text. The size of the created surface is just big enough to contain the text. The caller is responsible to release this surface (hw\_surface\_free) when it is no more needed.

void hw wait (int s delay)

Waits for the specified number of seconds. This tells the system to pause the current thread for the given amount of time.

void hw\_event\_wait\_next (struct ei\_event\_t \*event)

Lets this processus sleep until a new event is available.

ei\_bool\_t hw\_event\_post\_app (void \*user\_param)

Put an application-generated event on the event queue. This will cause hw\_event\_wait\_next to wake.

double hw\_now ()

Returns the number of seconds since the library was initialised by hw\_init. Can be used to measure elpased time between to calls.

#### **Variables**

- const int EI\_MOUSEBUTTON\_LEFT
- const int EI MOUSEBUTTON MIDDLE
- const int EI MOUSEBUTTON RIGHT

### 4.4.1 Detailed Description

Low level interface with the graphic hadware. This interface is based on the SDL library.

Created by François Bérard on 30.12.11. Copyright 2011 Ensimag. All rights reserved.

# 4.4.2 Function Documentation

4.4.2.1 ei surface thw\_create\_window( ei size t \* size, const ei bool t fullScreen)

Opens the main graphical window of the application.

#### **Parameters**

size	Number of horizontal and vertical pixels.
fullScreen	If true, opens the window in full screen. Otherwise opens a floating window.

#### Returns

The unlocked drawing surface (see hw\_surface\_lock). This surface should not be freed by calling hw\_\circ surface free, it is freed when releasing access to the low-level services by calling hw quit.

4.4.2.2 ei\_bool\_t hw\_event\_post\_app ( void \* user\_param )

Put an application-generated event on the event queue. This will cause hw\_event\_wait\_next to wake.

#### **Parameters**

|--|

4.4.2.3 void hw\_event\_wait\_next ( struct ei\_event\_t \* event )

Lets this processus sleep until a new event is available.

## **Parameters**

event	Where to store the new event. The structure must be allocated by the caller. On return, the structure is
	filled with informations about the new event.

4.4.2.4 ei\_color\_t hw\_get\_pixel ( const ei\_surface\_t surface, const ei\_point\_t pos )

Returns the color of the pixel at pos.

#### **Parameters**

surface	The surface on which to draw.
pos	The position at which to draw.
color	The color to draw.

# Returns

The color of teh pixel.

4.4.2.5 ei\_surface\_t hw\_image\_load ( const char \* filename )

Creates a surface and loads into it an image read from a file. The caller is responsible to release this surface (hw\_surface\_free) when it is no more needed.

#### **Parameters**

filename	The name of the file containing the image.	The file can be .bmp, .png, .jpg, .tga
----------	--	--

## Returns

A new unlocked surface containing the image.

4.4.2.6 double hw\_now ( )

Returns the number of seconds since the library was initialised by <a href="hw\_init">hw\_init</a>. Can be used to measure elpased time between to calls.

## Returns

The current time, in seconds.

4.4.2.7 void hw\_put\_pixel ( const ei\_surface\_t surface, const ei\_point\_t pos, const ei\_color\_t color )

Draw the pixel of the surface at pos.

#### **Parameters**

surface	The surface on which to draw.
pos	The position at which to draw.
color	The color to draw.

 $4.4.2.8 \quad ei\_surface\_t \ hw\_surface\_create \ ( \ const \ ei\_surface\_t \ \textit{root}, \ const \ ei\_size\_t * \textit{size} \ )$ 

Allocates an off-screen drawing surface.

### **Parameters**

root	The root window which channel indices will be used. This insures that the offscreen uses the same channel indices (Red, Green, Blue, Alpha) as the root surface.
size	Number of horizontal and vertical pixels.
force_alpha	If true, then the returned surface will use an alpha channel regardless of root having an alpha channel or not.

## Returns

The unlocked drawing surface (see hw\_surface\_lock). The surface should be freed by calling hw\_surface\_ cfree.

4.4.2.9 void hw\_surface\_free ( ei\_surface\_t surface )

Frees a surface allocated by hw\_surface\_create. This must be called on an unlocked surface (see hw\_surface\_cullock).

#### **Parameters**

surface The surf	face to be freed.
------------------	-------------------

4.4.2.10 ei\_rect\_t hw\_surface\_get\_rect ( const ei\_surface\_t surface )

Returns the rectangle of a surface (origin and size).

# **Parameters**

#### Returns

The rectangle of the surface.

4.4.2.11 ei\_size\_t hw\_surface\_get\_size ( const ei\_surface\_t surface )

Returns the size of a surface.

#### **Parameters**

surface The surface which size is requested
---

#### Returns

The size of the surface.

4.4.2.12 void hw\_surface\_lock ( ei\_surface\_t surface )

Gains exclusive access to a surface. Every call to this function must be matched by a call to hw\_surface\_unlock. The address of the pixel buffer may change while the surface is unlocked. Thus, hw\_surface\_get\_buffer must called after each call to this function.

#### **Parameters**

surface	The surface to lock.

4.4.2.13 void hw\_surface\_unlock ( ei\_surface\_t surface )

Releases the exclusive access to a surface that was locked by hw\_surface\_lock.

#### **Parameters**

e surface to unlock.	surface
----------------------	---------

4.4.2.14 void hw\_surface\_update\_rects ( const ei\_linked\_rect\_t \* rects )

Requests that a list of rectangular regions of the root surface be updated on screen.

#### **Parameters**

rects	The list of rectangle to be updated on screen. If NULL, then the entire surface is updated.
-------	---

4.4.2.15 void hw\_text\_compute\_size ( const char \* text, const ei\_font\_t font, int \* width, int \* height )

Computes the size of a text surface givent the font and the text.

#### **Parameters**

text	The string of the message.
font	The font used to render the text.
width,height	Addresses where to store the computed width and height of the text surface.

4.4.2.16 ei\_surface\_t hw\_text\_create\_surface ( const char \* text, const ei\_font\_t font, const ei\_color\_t \* color )

Creates a surface containing a text. The size of the created surface is just big enough to contain the text. The caller is responsible to release this surface (hw\_surface\_free) when it is no more needed.

#### **Parameters**

text	The string of the message.
font	The font used to render the text.
color	The text color. The alpha parameter is not used. However, the text is rendered with alpha blending to smooth the curves of the letters (anti-aliasing).

#### Returns

A newly created unlocked surface containing an anti-aliased rendering of the text. The anti-aliasing is implemented with the alpha channel of the surface: pixels on the text's boundaries have some transparency.

4.4.2.17 ei\_font\_t hw\_text\_font\_create ( const char \* filename, int size )

Creates a font that can be used to render text. The font must be freed by calling hw\_text\_font\_free.

## **Parameters**

filename	The path to the file containing the ttf font definition. Can be relative.
style	The style of the font (normal, bold,).
size	The size of the characters in pixels.

#### Returns

The font.

4.4.2.18 void hw\_text\_font\_free ( ei\_font\_t font )

Frees a font created by hw\_text\_font\_create.

# **Parameters**

font	The font to be freed.
------	-----------------------

4.4.2.19 void hw\_wait ( int s\_delay )

Waits for the specified number of seconds. This tells the system to pause the current thread for the given amount of time.

## **Parameters**

s_delay	The amount of time, in seconds, to wait.
---------	--

# Index

alpha	ei ev keyup
ei_color_t, 6	ei_event.h, 17
,	ei ev last
button_number	ei_event.h, 17
ei_mouse_event_t, 11	ei_ev_mouse_buttondown
	ei_event.h, 17
ei_anc_center	ei ev mouse buttonup
ei_types.h, 20	ei_event.h, 17
ei_anc_east	ei ev mouse move
ei_types.h, 20	ei_event.h, 17
ei_anc_none	ei_ev_none
ei_types.h, 20	ei_event.h, 16
ei_anc_north	ei_ev_touch_begin
ei_types.h, 20	ei_event.h, 17
ei_anc_northeast	ei_ev_touch_end
ei_types.h, 20	ei_event.h, 17
ei_anc_northwest	ei_ev_touch_move
ei_types.h, 20	ei event.h, 17
ei_anc_south	ei_event.h
ei_types.h, 20	ei_ev_app, 16
ei_anc_southeast	ei_ev_display, 17
ei_types.h, 20	ei_ev_keychar, 17
ei_anc_southwest	ei_ev_keydown, 17
ei_types.h, 20	ei_ev_keyup, 17
ei_anc_west	ei_ev_last, 17
ei_types.h, 20	ei_ev_mouse_buttondown, 17
ei_anchor_t	ei_ev_mouse_buttonup, 17
ei_types.h, 20	ei_ev_mouse_move, 17
ei_app_event_t, 5	ei_ev_none, 16
ei_axis_both	ei_ev_touch_begin, 17
ei_types.h, 20	ei_ev_touch_end, 17
ei_axis_none	ei_ev_touch_move, 17
ei_types.h, 20	ei_eventtype_t, 16
ei_axis_set_t	ei_mod_alt, 17
ei_types.h, 20	ei_mod_alt_grad, 17
ei_axis_x	ei mod ctrl, 17
ei_types.h, 20	ei_mod_cttl, 17 ei_mod_meta_left, 17
el_axis_y	ei mod meta right, 17
ei_types.h, 20	ei_mod_meta_ngnt, 17
ei_color_t, 5	ei_modifier_key_t, 17
alpha, 6	ei_event_t, 7
ei_display_event_t, 6	ei_eventtype_t
ei_ev_app	ei_event.h, 16
ei_event.h, 16 ei_ev_display	ei_font_t
	ei types.h, 20
ei_event.h, 17 ei ev keychar	ei_types.11, 20 ei_key_event_t, 7
ei_event.h, 17	ei_linked_point_t, 8
ei_ev_keydown	ei_linked_rect_t, 9
ei_event.h, 17	ei_linked_tag_t, 10
01_0V0111.11, 17	oi_minou_tay_t, iv

30 INDEX

ei_main	hw_interface.h, 22
ei_main.h, 17	hw_event_post_app
ei_main.h	hw_interface.h, 23
ei_main, 17	hw_event_wait_next
ei_mod_alt	hw_interface.h, 23
ei_event.h, 17	hw_get_pixel
ei_mod_alt_grad	hw_interface.h, 23
ei_event.h, 17	hw_image_load
ei_mod_ctrl	hw_interface.h, 23
ei_event.h, 17	hw_interface.h
ei_mod_meta_left	hw_create_window, 22
ei_event.h, 17	hw_event_post_app, 23
ei_mod_meta_right	hw_event_wait_next, 23
ei_event.h, 17	hw_get_pixel, 23
ei_mod_shift	hw_image_load, 23
ei_event.h, 17	hw_now, 24
ei_modifier_key_t	hw_put_pixel, 24
ei_event.h, 17	hw_surface_create, 24
ei_mouse_event_t, 10	hw_surface_free, 24
button_number, 11	hw_surface_get_rect, 25
ei_point_t, 11	hw_surface_get_size, 25
y, 11	hw_surface_lock, 25
ei_rect_t, 12	hw_surface_unlock, 25
ei_relief_none	hw_surface_update_rects, 26
ei_types.h, 20	hw_text_compute_size, 26
ei_relief_raised	hw_text_create_surface, 26
ei_types.h, 20	hw_text_font_create, 26
ei_relief_sunken	hw_text_font_free, 28
ei_types.h, 20	hw_wait, 28
ei_relief_t	hw_now
ei_types.h, 20	hw_interface.h, 24
ei_size_t, 12	hw_put_pixel
ei_touch_event_t, 13	hw_interface.h, 24
primary, 14	hw_surface_create
touch_id, 14	hw_interface.h, 24
ei_types.h	hw_surface_free
ei_anc_center, 20	hw_interface.h, 24
ei_anc_east, 20	hw_surface_get_rect
ei_anc_none, 20	hw_interface.h, 25
ei_anc_north, 20	hw_surface_get_size
ei_anc_northeast, 20	hw_interface.h, 25
ei_anc_northwest, 20	hw_surface_lock
ei_anc_south, 20	hw_interface.h, 25
ei_anc_southeast, 20	hw_surface_unlock
ei_anc_southwest, 20	hw_interface.h, 25
ei_anc_west, 20	hw_surface_update_rects
ei_anchor_t, 20	hw_interface.h, 26
ei_axis_both, 20	hw_text_compute_size
ei_axis_none, 20	hw_interface.h, 26
ei_axis_set_t, 20	hw_text_create_surface
ei_axis_x, 20	hw_interface.h, 26
ei_axis_y, 20	hw_text_font_create
ei_font_t, 20	hw_interface.h, 26
ei_relief_none, 20	hw_text_font_free
ei_relief_raised, 20	hw_interface.h, 28
ei_relief_sunken, 20	hw_wait
ei_relief_t, 20	hw_interface.h, 28
hw_create_window	include/ei_event.h, 15
LINE LICEUS VVIIILIVV	iiioiuuo/oi_ovoiit.ii, i∪

INDEX 31

```
include/ei_main.h, 17
include/ei_types.h, 18
include/hw_interface.h, 21

primary
ei_touch_event_t, 14

touch_id
ei_touch_event_t, 14

y
ei_point_t, 11
```