# GaeGebra

v0.1

Generated by Doxygen 1.10.0

# 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 _UIDropdownItem Struct Reference

**Public Attributes**

- UIElement base
- UIDropdownList ∗ parent_dropdown
- Sint32 dropdown_index
- char text [UITEXT_MAX_LENGTH+1]
- MouseState mouse_state

### 3.1.1 Member Data Documentation

#### 3.1.1.1 base

```
UIElement _UIDropdownItem::base
```

#### 3.1.1.2 dropdown_index

```
Sint32 _UIDropdownItem::dropdown_index
```

#### 3.1.1.3 mouse_state

```
MouseState _UIDropdownItem::mouse_state
```

#### 3.1.1.4 parent_dropdown

```
UIDropdownList∗ _UIDropdownItem::parent_dropdown
```

#### 3.1.1.5 text

```
char _UIDropdownItem::text[UITEXT_MAX_LENGTH+1]
```

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

- src/ui/ui_element/ui_element.c

## 3.2 _UISplitButtonItem Struct Reference

**Public Attributes**

- UIElement base
- UISplitButton ∗ parent_splitbutton
- Sint32 splitbutton_index
- char text [UITEXT_MAX_LENGTH+1]
- MouseState mouse_state

### 3.2.1 Member Data Documentation

#### 3.2.1.1 base

```
UIElement _UISplitButtonItem::base
```

#### 3.2.1.2 mouse_state

```
MouseState _UISplitButtonItem::mouse_state
```

#### 3.2.1.3 parent_splitbutton

```
UISplitButton* _UISplitButtonItem::parent_splitbutton
```

#### 3.2.1.4 splitbutton_index

```
Sint32 _UISplitButtonItem::splitbutton_index
```

#### 3.2.1.5 text

```
char _UISplitButtonItem::text[UITEXT_MAX_LENGTH+1]
```

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

- src/ui/ui_element/ui_element.c

## 3.3 AppData Struct Reference

```
#include <app.h>
```

### Public Attributes

- Vector ∗ windows
- Uint32 target_frame_time
- Uint32 last_frame_start
- Uint32 frame_start
- double delta_time

### 3.3.1 Member Data Documentation

#### 3.3.1.1 delta_time

```
double AppData::delta_time
```

#### 3.3.1.2 frame_start

```
Uint32 AppData::frame_start
```

#### 3.3.1.3 last_frame_start

```
Uint32 AppData::last_frame_start
```

#### 3.3.1.4 target_frame_time

```
Uint32 AppData::target_frame_time
```

#### 3.3.1.5 windows

```
Vector* AppData::windows
```

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

- src/app/app.h

## 3.4 Circle Struct Reference

```
#include <shape.h>
```

### Public Attributes

- IShape base
- Point ∗ center
- Point ∗ perimeter_point

## 3.4.1 Member Data Documentation

### 3.4.1.1 base

```
IShape Circle::base
```

### 3.4.1.2 center

```
Point* Circle::center
```

### 3.4.1.3 perimeter_point

```
Point* Circle::perimeter_point
```

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

- src/geometry/shape/shape.h

# 3.5 CoordinateSystem Struct Reference

```
#include <coordinate_system.h>
```

### Public Attributes

- Vector2 position
- Vector2 size
- Vector2 origin
- double zoom
- Vector ∗ shapes

## 3.5.1 Member Data Documentation

### 3.5.1.1 origin

```
Vector2 CoordinateSystem::origin
```

#### 3.5.1.2 position

```
Vector2 CoordinateSystem::position
```

#### 3.5.1.3 shapes

```
Vector* CoordinateSystem::shapes
```

#### 3.5.1.4 size

```
Vector2 CoordinateSystem::size
```

#### 3.5.1.5 zoom

```
double CoordinateSystem::zoom
```

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

- src/geometry/coordinate_system/coordinate_system.h

## 3.6 Font Struct Reference

```
#include <font.h>
```

**Public Attributes**

- TTF_Font * font
- int size

### 3.6.1 Member Data Documentation

#### 3.6.1.1 font

```
TTF_Font* Font::font
```

#### 3.6.1.2 size

```
int Font::size
```

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

- src/font/font.h

## 3.7 InputData Struct Reference

```
#include <input.h>
```

**Public Attributes**

- bool current_mouse_button_state [5]
- bool old_mouse_button_state [5]
- SDL_Point current_mouse_position
- SDL_Point old_mouse_position
- int mouse_wheel_delta
- Uint8 ∗ current_keyboard_state
- Uint8 ∗ old_keyboard_state
- int key_count

### 3.7.1 Member Data Documentation

#### 3.7.1.1 current_keyboard_state

```
Uint8* InputData::current_keyboard_state
```

#### 3.7.1.2 current_mouse_button_state

```
bool InputData::current_mouse_button_state[5]
```

#### 3.7.1.3 current_mouse_position

```
SDL_Point InputData::current_mouse_position
```

#### 3.7.1.4 key_count

```
int InputData::key_count
```

#### 3.7.1.5 mouse_wheel_delta

```
int InputData::mouse_wheel_delta
```

#### 3.7.1.6 old_keyboard_state

```
Uint8* InputData::old_keyboard_state
```

#### 3.7.1.7 old_mouse_button_state

```
bool InputData::old_mouse_button_state[5]
```

#### 3.7.1.8 old_mouse_position

```
SDL_Point InputData::old_mouse_position
```

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

- src/input/input.h

## 3.8 IShape Struct Reference

```
#include <shape.h>
```

### Public Attributes

- void(∗ draw )(CoordinateSystem ∗cs, IShape ∗self)
- void(∗ translate )(CoordinateSystem ∗cs, IShape ∗self, Vector2 translation)
- void(∗ destroy )(CoordinateSystem ∗cs, IShape ∗self)
- bool(∗ overlap_point )(CoordinateSystem ∗cs, IShape ∗self, Vector2 point)
- bool(∗ is_defined_by )(IShape ∗self, IShape ∗shape)

### 3.8.1 Member Data Documentation

#### 3.8.1.1 destroy

```
void(* IShape::destroy) (CoordinateSystem *cs, IShape *self)
```

#### 3.8.1.2 draw

```
void(* IShape::draw) (CoordinateSystem *cs, IShape *self)
```

#### 3.8.1.3 is_defined_by

```
bool(* IShape::is_defined_by) (IShape *self, IShape *shape)
```

#### 3.8.1.4 overlap_point

```
bool(* IShape::overlap_point) (CoordinateSystem *cs, IShape *self, Vector2 point)
```

#### 3.8.1.5 translate

```
void(* IShape::translate) (CoordinateSystem *cs, IShape *self, Vector2 translation)
```

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

- src/geometry/shape/shape.h

# 3.9 Line Struct Reference

```
#include <shape.h>
```

### Public Attributes

- IShape base
- Point ∗ p1
- Point ∗ p2

## 3.9.1 Member Data Documentation

### 3.9.1.1 base

```
IShape Line::base
```

### 3.9.1.2 p1

```
Point* Line::p1
```

### 3.9.1.3 p2

```
Point * Line::p2
```

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

- src/geometry/shape/shape.h

# 3.10 Point Struct Reference

```
#include <shape.h>
```

### Public Attributes

- IShape base
- Vector2 coordinates

## 3.10.1 Member Data Documentation

### 3.10.1.1 base

```
IShape Point::base
```

#### 3.10.1.2 coordinates

```
Vector2 Point::coordinates
```

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

- src/geometry/shape/shape.h

## 3.11 Texture Struct Reference

```
#include <texture.h>
```

**Public Attributes**

- SDL_Texture ∗ texture
- int width
- int height

### 3.11.1 Member Data Documentation

#### 3.11.1.1 height

```
int Texture::height
```

#### 3.11.1.2 texture

```
SDL_Texture* Texture::texture
```

#### 3.11.1.3 width

```
int Texture::width
```

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

- src/texture/texture.h

## 3.12 UIButton Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- char text [UITEXT_MAX_LENGTH+1]
- SDL_Point text_position
- Color color
- Color text_color
- Uint32 corner_radius
- MouseState mouse_state
- void(∗ on_click )(UIButton ∗self)

## 3.12.1 Member Data Documentation

### 3.12.1.1 base

```
UIElement UIButton::base
```

### 3.12.1.2 color

```
Color UIButton::color
```

### 3.12.1.3 corner_radius

```
Uint32 UIButton::corner_radius
```

### 3.12.1.4 mouse_state

```
MouseState UIButton::mouse_state
```

### 3.12.1.5 on_click

```
void(* UIButton::on_click) (UIButton *self)
```

### 3.12.1.6 text

```
char UIButton::text[UITEXT_MAX_LENGTH+1]
```

### 3.12.1.7 text_color

```
Color UIButton::text_color
```

#### 3.12.1.8 text_position

```
SDL_Point UIButton::text_position
```

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

- src/ui/ui_element/ui_element.h

## 3.13 UICheckbox Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- bool checked
- Color checked_color
- Color unchecked_color
- Uint32 corner_radius
- MouseState mouse_state
- void(∗ on_checked_changed )(UICheckbox ∗self, bool checked)

### 3.13.1 Member Data Documentation

#### 3.13.1.1 base

```
UIElement UICheckbox::base
```

#### 3.13.1.2 checked

```
bool UICheckbox::checked
```

#### 3.13.1.3 checked_color

```
Color UICheckbox::checked_color
```

#### 3.13.1.4 corner_radius

```
Uint32 UICheckbox::corner_radius
```

#### 3.13.1.5 mouse_state

```
MouseState UICheckbox::mouse_state
```

#### 3.13.1.6 on_checked_changed

```
void(* UICheckbox::on_checked_changed) (UICheckbox *self, bool checked)
```

#### 3.13.1.7 unchecked_color

```
Color UICheckbox::unchecked_color
```

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

- src/ui/ui_element/ui_element.h

## 3.14 UIConstraint Struct Reference

```
#include <ui_constraint.h>
```

**Public Attributes**

- double value
- ConstraintType constraint_type
- void(* recalculate )(void *self)

### 3.14.1 Member Data Documentation

#### 3.14.1.1 constraint_type

```
ConstraintType UIConstraint::constraint_type
```

#### 3.14.1.2 recalculate

```
void(* UIConstraint::recalculate) (void *self)
```

#### 3.14.1.3 value

```
double UIConstraint::value
```

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

- src/ui/ui_constraint/ui_constraint.h

## 3.15 UIConstraints Struct Reference

```
#include <ui_constraint.h>
```

**Public Attributes**

- UIConstraint x
- UIConstraint y
- UIConstraint width
- UIConstraint height

## 3.15.1 Member Data Documentation

### 3.15.1.1 height

```
UIConstraint UIConstraints::height
```

### 3.15.1.2 width

```
UIConstraint UIConstraints::width
```

### 3.15.1.3 x

```
UIConstraint UIConstraints::x
```

### 3.15.1.4 y

```
UIConstraint UIConstraints::y
```

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

- src/ui/ui_constraint/ui_constraint.h

# 3.16 UIContainer Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- Vector ∗ children
- void(∗ on_size_changed )(UIContainer ∗self, SDL_Point size)

## 3.16.1 Member Data Documentation

### 3.16.1.1 base

```
UIElement UIContainer::base
```

#### 3.16.1.2 children

```
Vector* UIContainer::children
```

#### 3.16.1.3 on_size_changed

```
void(* UIContainer::on_size_changed) (UIContainer *self, SDL_Point size)
```

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

- src/ui/ui_element/ui_element.h

## 3.17 UIData Struct Reference

```
#include <ui.h>
```

**Public Attributes**

- UIContainer ∗ main_container
- char text_input [SDL_TEXTINPUTEVENT_TEXT_SIZE]
- bool backspace_pressed
- bool mouse_captured
- UISplitButton ∗ expanded_splitbutton

### 3.17.1 Member Data Documentation

#### 3.17.1.1 backspace_pressed

```
bool UIData::backspace_pressed
```

#### 3.17.1.2 expanded_splitbutton

```
UISplitButton* UIData::expanded_splitbutton
```

#### 3.17.1.3 main_container

```
UIContainer* UIData::main_container
```

#### 3.17.1.4 mouse_captured

```
bool UIData::mouse_captured
```

#### 3.17.1.5 text_input

```
char UIData::text_input[SDL_TEXTINPUTEVENT_TEXT_SIZE]
```

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

- src/ui/ui.h

## 3.18 UIDropdownList Struct Reference

```
#include <ui_element.h>
```

### Public Attributes

- UIElement base
- Vector ∗ items
- Uint32 selected_item
- bool expanded
- Color color
- Color text_color
- Uint32 corner_radius
- void(∗ on_selection_changed )(UIDropdownList ∗self, Sint32 index)

### 3.18.1 Member Data Documentation

#### 3.18.1.1 base

```
UIElement UIDropdownList::base
```

#### 3.18.1.2 color

```
Color UIDropdownList::color
```

#### 3.18.1.3 corner_radius

```
Uint32 UIDropdownList::corner_radius
```

#### 3.18.1.4 expanded

```
bool UIDropdownList::expanded
```

#### 3.18.1.5 items

```
Vector* UIDropdownList::items
```

#### 3.18.1.6 on_selection_changed

```
void(∗ UIDropdownList::on_selection_changed) (UIDropdownList ∗self, Sint32 index)
```

#### 3.18.1.7 selected_item

```
Uint32 UIDropdownList::selected_item
```

#### 3.18.1.8 text_color

```
Color UIDropdownList::text_color
```

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

- src/ui/ui_element/ui_element.h

## 3.19 UIElement Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement ∗ parent
- UIConstraints constraints
- SDL_Point position
- SDL_Point size
- void(∗ update )(UIElement ∗self)
- void(∗ recalculate )(UIElement ∗sibling, UIElement ∗self)
- void(∗ render )(UIElement ∗self)
- void(∗ destroy )(UIElement ∗self)

### 3.19.1 Member Data Documentation

#### 3.19.1.1 constraints

```
UIConstraints UIElement::constraints
```

#### 3.19.1.2 destroy

```
void(∗ UIElement::destroy) (UIElement ∗self)
```

#### 3.19.1.3 parent

```
UIElement∗ UIElement::parent
```

#### 3.19.1.4 position

```
SDL_Point UIElement::position
```

#### 3.19.1.5 recalculate

```
void(* UIElement::recalculate) (UIElement *sibling, UIElement *self)
```

#### 3.19.1.6 render

```
void(* UIElement::render) (UIElement *self)
```

#### 3.19.1.7 size

```
SDL_Point UIElement::size
```

#### 3.19.1.8 update

```
void(* UIElement::update) (UIElement *self)
```

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

- src/ui/ui_element/ui_element.h

## 3.20 UIImageButton Struct Reference

```
#include <ui_element.h>
```

#### Public Attributes

- UIElement base
- Texture ∗ texture
- MouseState mouse_state
- void(∗ on_click )(UIImageButton ∗self)

### 3.20.1 Member Data Documentation

#### 3.20.1.1 base

```
UIElement UIImageButton::base
```

#### 3.20.1.2 mouse_state

```
MouseState UIImageButton::mouse_state
```

#### 3.20.1.3 on_click

```
void(* UIImageButton::on_click) (UIImageButton *self)
```

#### 3.20.1.4 texture

```
Texture* UIImageButton::texture
```

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

- src/ui/ui_element/ui_element.h

## 3.21 UILabel Struct Reference

```
#include <ui_element.h>
```

### Public Attributes

- UIElement base
- char text [UITEXT_MAX_LENGTH+1]
- Color color

### 3.21.1 Member Data Documentation

#### 3.21.1.1 base

```
UIElement UILabel::base
```

#### 3.21.1.2 color

```
Color UILabel::color
```

#### 3.21.1.3 text

```
char UILabel::text[UITEXT_MAX_LENGTH+1]
```

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

- src/ui/ui_element/ui_element.h

## 3.22 UIPanel Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- Color color
- Color border_color
- Uint32 border_width
- Uint32 corner_radius

### 3.22.1 Member Data Documentation

#### 3.22.1.1 base

```
UIElement UIPanel::base
```

#### 3.22.1.2 border_color

```
Color UIPanel::border_color
```

#### 3.22.1.3 border_width

```
Uint32 UIPanel::border_width
```

#### 3.22.1.4 color

```
Color UIPanel::color
```

#### 3.22.1.5 corner_radius

```
Uint32 UIPanel::corner_radius
```

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

- src/ui/ui_element/ui_element.h

## 3.23 UISlider Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- double value
- Color color
- Color slider_color
- Uint32 thickness
- Uint32 corner_radius
- MouseState mouse_state
- void(∗ on_value_changed )(UISlider ∗self, double value)

## 3.23.1 Member Data Documentation

### 3.23.1.1 base

```
UIElement UISlider::base
```

### 3.23.1.2 color

```
Color UISlider::color
```

### 3.23.1.3 corner_radius

```
Uint32 UISlider::corner_radius
```

### 3.23.1.4 mouse_state

```
MouseState UISlider::mouse_state
```

### 3.23.1.5 on_value_changed

```
void(* UISlider::on_value_changed) (UISlider *self, double value)
```

### 3.23.1.6 slider_color

```
Color UISlider::slider_color
```

### 3.23.1.7 thickness

```
Uint32 UISlider::thickness
```

#### 3.23.1.8 value

```
double UISlider::value
```

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

- src/ui/ui_element/ui_element.h

## 3.24 UISplitButton Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- Vector ∗ items
- bool expanded
- Color color
- Color text_color
- Uint32 corner_radius
- void(∗ on_item_clicked )(UISplitButton ∗self, Sint32 index)
- bool auto_dropdown

### 3.24.1 Member Data Documentation

#### 3.24.1.1 auto_dropdown

```
bool UISplitButton::auto_dropdown
```

#### 3.24.1.2 base

```
UIElement UISplitButton::base
```

#### 3.24.1.3 color

```
Color UISplitButton::color
```

#### 3.24.1.4 corner_radius

```
Uint32 UISplitButton::corner_radius
```

#### 3.24.1.5 expanded

```
bool UISplitButton::expanded
```

#### 3.24.1.6 items

```
Vector* UISplitButton::items
```

#### 3.24.1.7 on_item_clicked

```
void(* UISplitButton::on_item_clicked) (UISplitButton *self, Sint32 index)
```

#### 3.24.1.8 text_color

```
Color UISplitButton::text_color
```

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

- src/ui/ui_element/ui_element.h

## 3.25 UITextbox Struct Reference

```
#include <ui_element.h>
```

**Public Attributes**

- UIElement base
- char text [UITEXT_MAX_LENGTH+1]
- Color color
- Color text_color
- Uint32 corner_radius
- bool focused
- MouseState mouse_state
- void(* on_text_changed )(UITextbox *self, const char *text)

### 3.25.1 Member Data Documentation

#### 3.25.1.1 base

```
UIElement UITextbox::base
```

#### 3.25.1.2 color

```
Color UITextbox::color
```

#### 3.25.1.3 corner_radius

```
Uint32 UITextbox::corner_radius
```

#### 3.25.1.4 focused

```
bool UITextbox::focused
```

#### 3.25.1.5 mouse_state

```
MouseState UITextbox::mouse_state
```

#### 3.25.1.6 on_text_changed

```
void(* UITextbox::on_text_changed) (UITextbox *self, const char *text)
```

#### 3.25.1.7 text

```
char UITextbox::text[UITEXT_MAX_LENGTH+1]
```

#### 3.25.1.8 text_color

```
Color UITextbox::text_color
```

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

- src/ui/ui_element/ui_element.h

## 3.26 Vector Struct Reference

```
#include <vector.h>
```

**Public Attributes**

- size_t capacity
- size_t size
- void ** data

### 3.26.1 Member Data Documentation

#### 3.26.1.1 capacity

```
size_t Vector::capacity
```

#### 3.26.1.2 data

```
void** Vector::data
```

#### 3.26.1.3 size

```
size_t Vector::size
```

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

- src/utils/vector/vector.h

## 3.27 Vector2 Struct Reference

```
#include <vector2.h>
```

**Public Attributes**

- double x
- double y

### 3.27.1 Member Data Documentation

#### 3.27.1.1 x

```
double Vector2::x
```

#### 3.27.1.2 y

```
double Vector2::y
```

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

- src/geometry/vector2/vector2.h

## 3.28 Window Struct Reference

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

## Public Attributes

- SDL_Window ∗ window
- SDL_Renderer ∗ renderer
- InputData input_data
- UIData ui_data
- bool close_requested

## 3.28.1 Member Data Documentation

### 3.28.1.1 close_requested

```
bool Window::close_requested
```

### 3.28.1.2 input_data

```
InputData Window::input_data
```

### 3.28.1.3 renderer

```
SDL_Renderer∗ Window::renderer
```

### 3.28.1.4 ui_data

```
UIData Window::ui_data
```

### 3.28.1.5 window

```
SDL_Window∗ Window::window
```

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

- src/window/window.h

# Chapter 4

# File Documentation

## 4.1 src/app/app.c File Reference

```
#include "app.h"
#include "../window/window.h"
#include "../renderer/renderer.h"
#include "../input/input.h"
#include "../ui/ui.h"
```

#### Functions

- void app_init ()
- void app_update ()
- void app_render ()
- void app_request_close ()
- void app_close ()
- void app_set_target_fps (Uint32 fps)
- void app_set_target (Window ∗window)
- Vector ∗ app_get_windows ()
- double app_get_time ()
- double app_get_delta_time ()
- void _app_add_window (Window ∗window)

#### Variables

- AppData app_data

### 4.1.1 Function Documentation

#### 4.1.1.1 _app_add_window()

```
void _app_add_window (
            Window ∗ window )
```

#### 4.1.1.2 app_close()

```
void app_close ( )
```

#### 4.1.1.3 app_get_delta_time()

```
double app_get_delta_time ( )
```

#### 4.1.1.4 app_get_time()

```
double app_get_time ( )
```

#### 4.1.1.5 app_get_windows()

```
Vector * app_get_windows ( )
```

#### 4.1.1.6 app_init()

```
void app_init ( )
```

#### 4.1.1.7 app_render()

```
void app_render ( )
```

#### 4.1.1.8 app_request_close()

```
void app_request_close ( )
```

#### 4.1.1.9 app_set_target()

```
void app_set_target (
             Window * window )
```

#### 4.1.1.10 app_set_target_fps()

```
void app_set_target_fps (
             Uint32 fps )
```

#### 4.1.1.11 app_update()

```
void app_update ( )
```

### 4.1.2 Variable Documentation

#### 4.1.2.1 app_data

```
AppData app_data
```

## 4.2 src/app/app.h File Reference

```
#include "../window/window.h"
#include "../utils/vector/vector.h"
```

#### Classes

- struct AppData

#### Typedefs

- typedef struct AppData AppData

#### Functions

- void app_init ()
- void app_update ()
- void app_render ()
- void app_request_close ()
- void app_close ()
- void app_set_target_fps (Uint32 fps)
- void app_set_target (Window ∗window)
- Vector ∗ app_get_windows ()
- double app_get_time ()
- double app_get_delta_time ()
- void _app_add_window (Window ∗window)

### 4.2.1 Typedef Documentation

#### 4.2.1.1 AppData

```
typedef struct AppData AppData
```

### 4.2.2 Function Documentation

#### 4.2.2.1 _app_add_window()

```
void _app_add_window (
            Window * window )
```

#### 4.2.2.2 app_close()

```
void app_close ( )
```

#### 4.2.2.3 app_get_delta_time()

```
double app_get_delta_time ( )
```

#### 4.2.2.4 app_get_time()

```
double app_get_time ( )
```

#### 4.2.2.5 app_get_windows()

```
Vector * app_get_windows ( )
```

#### 4.2.2.6 app_init()

```
void app_init ( )
```

#### 4.2.2.7 app_render()

```
void app_render ( )
```

#### 4.2.2.8 app_request_close()

```
void app_request_close ( )
```

#### 4.2.2.9 app_set_target()

```
void app_set_target (
             Window * window )
```

#### 4.2.2.10 app_set_target_fps()

```
void app_set_target_fps (
             Uint32 fps )
```

#### 4.2.2.11 app_update()

```
void app_update ( )
```

## 4.3 app.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
#endif

#include "../window/window.h"
#include "../utils/vector/vector.h"

typedef struct AppData
{
    Vector* windows;
    Uint32 target_frame_time;
    Uint32 last_frame_start;
    Uint32 frame_start;
    double delta_time;
} AppData;

void app_init();
void app_update();
void app_render();
void app_request_close();
void app_close();
void app_set_target_fps(Uint32 fps);

void app_set_target(Window* window);
Vector* app_get_windows();
double app_get_time();
double app_get_delta_time();

//internal functions
void _app_add_window(Window* window);
```

## 4.4 src/color/color.c File Reference

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

### Functions

- Color color_from_hex (int hex)
- Color color_from_rgb (int r, int g, int b)
- Color color_from_rgba (int r, int g, int b, int a)
- Color color_from_hsv (double h, double s, double v)
- Color color_from_grayscale (int value)
- Color color_fade (Color color, double fade)
- Color color_shift (Color color, int shift)
- Color color_clever_shift (Color color, int shift)

### 4.4.1 Function Documentation

#### 4.4.1.1 color_clever_shift()

```
Color color_clever_shift (
              Color color,
              int shift )
```

#### 4.4.1.2 color_fade()

```
Color color_fade (
            Color color,
            double fade )
```

#### 4.4.1.3 color_from_grayscale()

```
Color color_from_grayscale (
            int value )
```

#### 4.4.1.4 color_from_hex()

```
Color color_from_hex (
            int hex )
```

#### 4.4.1.5 color_from_hsv()

```
Color color_from_hsv (
            double h,
            double s,
            double v )
```

#### 4.4.1.6 color_from_rgb()

```
Color color_from_rgb (
            int r,
            int g,
            int b )
```

#### 4.4.1.7 color_from_rgba()

```
Color color_from_rgba (
            int r,
            int g,
            int b,
            int a )
```

#### 4.4.1.8 color_shift()

```
Color color_shift (
            Color color,
            int shift )
```

## 4.5 src/color/color.h File Reference

### Macros

- #define WHITE (Color) { 255, 255, 255, 255 }
- #define BLACK (Color) { 0, 0, 0, 255 }
- #define GRAY (Color) { 128, 128, 128, 255 }
- #define DARK_GRAY (Color) { 40, 40, 40, 255 }
- #define RED (Color) { 255, 0, 0, 255 }
- #define GREEN (Color) { 0, 255, 0, 255 }
- #define BLUE (Color) { 0, 0, 255, 255 }
- #define YELLOW (Color) { 255, 255, 0, 255 }
- #define MAGENTA (Color) { 255, 0, 255, 255 }
- #define CYAN (Color) { 0, 255, 255, 255 }
- #define TRANSPARENT (Color) { 0, 0, 0, 0 }

### Typedefs

- typedef SDL_Color Color

### Functions

- Color color_from_hex (int hex)
- Color color_from_rgb (int r, int g, int b)
- Color color_from_rgba (int r, int g, int b, int a)
- Color color_from_hsv (double h, double s, double v)
- Color color_from_grayscale (int value)
- Color color_fade (Color color, double fade)
- Color color_shift (Color color, int shift)
- Color color_clever_shift (Color color, int shift)

### 4.5.1 Macro Definition Documentation

#### 4.5.1.1 BLACK

```
#define BLACK (Color) { 0, 0, 0, 255 }
```

#### 4.5.1.2 BLUE

```
#define BLUE (Color) { 0, 0, 255, 255 }
```

#### 4.5.1.3 CYAN

```
#define CYAN (Color) { 0, 255, 255, 255 }
```

#### 4.5.1.4 DARK_GRAY

```
#define DARK_GRAY (Color) { 40, 40, 40, 255 }
```

#### 4.5.1.5 GRAY

```
#define GRAY (Color) { 128, 128, 128, 255 }
```

#### 4.5.1.6 GREEN

```
#define GREEN (Color) { 0, 255, 0, 255 }
```

#### 4.5.1.7 MAGENTA

```
#define MAGENTA (Color) { 255, 0, 255, 255 }
```

#### 4.5.1.8 RED

```
#define RED (Color) { 255, 0, 0, 255 }
```

#### 4.5.1.9 TRANSPARENT

```
#define TRANSPARENT (Color) { 0, 0, 0, 0 }
```

#### 4.5.1.10 WHITE

```
#define WHITE (Color) { 255, 255, 255, 255 }
```

#### 4.5.1.11 YELLOW

```
#define YELLOW (Color) { 255, 255, 0, 255 }
```

### 4.5.2 Typedef Documentation

#### 4.5.2.1 Color

```
typedef SDL_Color Color
```

## 4.5.3 Function Documentation

### 4.5.3.1 color_clever_shift()

```
Color color_clever_shift (
            Color color,
            int shift )
```

### 4.5.3.2 color_fade()

```
Color color_fade (
            Color color,
            double fade )
```

### 4.5.3.3 color_from_grayscale()

```
Color color_from_grayscale (
            int value )
```

### 4.5.3.4 color_from_hex()

```
Color color_from_hex (
            int hex )
```

### 4.5.3.5 color_from_hsv()

```
Color color_from_hsv (
            double h,
            double s,
            double v )
```

### 4.5.3.6 color_from_rgb()

```
Color color_from_rgb (
            int r,
            int g,
            int b )
```

### 4.5.3.7 color_from_rgba()

```
Color color_from_rgba (
            int r,
            int g,
            int b,
            int a )
```

#### 4.5.3.8 color_shift()

```
Color color_shift (
              Color color,
              int shift )
```

## 4.6 color.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
#endif

typedef SDL_Color Color;

#define WHITE (Color) { 255, 255, 255, 255 }
#define BLACK (Color) { 0, 0, 0, 255 }
#define GRAY (Color) { 128, 128, 128, 255 }
#define DARK_GRAY (Color) { 40, 40, 40, 255 }
#define RED (Color) { 255, 0, 0, 255 }
#define GREEN (Color) { 0, 255, 0, 255 }
#define BLUE (Color) { 0, 0, 255, 255 }
#define YELLOW (Color) { 255, 255, 0, 255 }
#define MAGENTA (Color) { 255, 0, 255, 255 }
#define CYAN (Color) { 0, 255, 255, 255 }
#define TRANSPARENT (Color) { 0, 0, 0, 0 }

Color color_from_hex(int hex);
Color color_from_rgb(int r, int g, int b);
Color color_from_rgba(int r, int g, int b, int a);
Color color_from_hsv(double h, double s, double v);
Color color_from_grayscale(int value);
Color color_fade(Color color, double fade);
Color color_shift(Color color, int shift);
Color color_clever_shift(Color color, int shift);
```

## 4.7 src/font/font.c File Reference

```
#include "font.h"
#include "../utils/vector/vector.h"
```

### Functions

- Font ∗ font_load (const char ∗path, int size)
- void _font_init ()
- void _font_close ()

### Variables

- Vector ∗ fonts

## 4.7.1 Function Documentation

### 4.7.1.1 _font_close()

```
void _font_close ( )
```

### 4.7.1.2 _font_init()

```
void _font_init ( )
```

### 4.7.1.3 font_load()

```
Font * font_load (
             const char * path,
             int size )
```

## 4.7.2 Variable Documentation

### 4.7.2.1 fonts

```
Vector* fonts
```

# 4.8 src/font/font.h File Reference

### Classes

- struct Font

### Typedefs

- typedef struct Font Font

### Functions

- Font ∗ font_load (const char ∗path, int size)
- void _font_init ()
- void _font_close ()

## 4.8.1 Typedef Documentation

### 4.8.1.1 Font

```
typedef struct Font Font
```

## 4.8.2 Function Documentation

### 4.8.2.1 _font_close()

```
void _font_close ( )
```

### 4.8.2.2 _font_init()

```
void _font_init ( )
```

### 4.8.2.3 font_load()

```
Font * font_load (
              const char * path,
              int size )
```

## 4.9 font.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL_ttf.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL_ttf.h>
#endif

typedef struct Font
{
    TTF_Font* font;
    int size;
} Font;

Font* font_load(const char* path, int size);

//internal functions
void _font_init();
void _font_close();
```

## 4.10 src/geometry/coordinate_system/coordinate_system.c File Reference

```
#include "coordinate_system.h"
#include "../../renderer/renderer.h"
#include "../../utils/math/math.h"
```

## Functions

- CoordinateSystem ∗ coordinate_system_create (Vector2 position, Vector2 size, Vector2 origin)

  *Creates a coordinate system.*
- void coordinate_system_destroy (CoordinateSystem ∗cs)

  *Destroys a coordinate system.*
- Vector2 screen_to_coordinates (CoordinateSystem ∗cs, Vector2 point)

  *Translates a point from the screen to the coordinate system.*
- Vector2 coordinates_to_screen (CoordinateSystem ∗cs, Vector2 point)

  *Translates a point from the coordinate system to the screen.*
- bool coordinate_system_is_hovered (CoordinateSystem ∗cs, Vector2 point)

  *Returns whether the coordinate system is hovered by the point.*
- IShape ∗ coordinate_system_get_hovered_shape (CoordinateSystem ∗cs, Vector2 point)

  *Returns the shape hovered by the point.*
- void coordinate_system_translate (CoordinateSystem ∗cs, Vector2 translation)

  *Translates the coordinate system.*
- void coordinate_system_zoom (CoordinateSystem ∗cs, double zoom)

  *Zooms into the coordinate system.*
- void coordinate_system_update_dimensions (CoordinateSystem ∗cs, Vector2 position, Vector2 size)

  *Updates the dimensions of the coordinate system.*
- void coordinate_system_draw (CoordinateSystem ∗cs)

  *Draws the coordinate system.*

## 4.10.1 Function Documentation

### 4.10.1.1 coordinate_system_create()

```
CoordinateSystem * coordinate_system_create (
            Vector2 position,
            Vector2 size,
            Vector2 origin )
```

Creates a coordinate system.

**Parameters**

| | |
|---|---|
| *position* | The position of the coordinate system in the screen |
| *size* | The size of the coordinate system (in pixels) |
| *origin* | The origin of the coordinate system (relative to the coordinate system (normalized)) |

**Returns**

CoordinateSystem∗ The created coordinate system

### 4.10.1.2 coordinate_system_destroy()

```
void coordinate_system_destroy (
            CoordinateSystem * cs )
```

Destroys a coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to destroy |

#### 4.10.1.3 coordinate_system_draw()

```
void coordinate_system_draw (
            CoordinateSystem * cs )
```

Draws the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to draw |

#### 4.10.1.4 coordinate_system_get_hovered_shape()

```
IShape * coordinate_system_get_hovered_shape (
            CoordinateSystem * cs,
            Vector2 point )
```

Returns the shape hovered by the point.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to check |
| *point* | The point to check |

**Returns**

IShape∗ The hovered shape (NULL if none)

#### 4.10.1.5 coordinate_system_is_hovered()

```
bool coordinate_system_is_hovered (
            CoordinateSystem * cs,
            Vector2 point )
```

Returns whether the coordinate system is hovered by the point.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to check |
| *point* | The point to check |

#### 4.10.1.6 coordinate_system_translate()

```
void coordinate_system_translate (
            CoordinateSystem * cs,
            Vector2 translation )
```

Translates the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to translate |
| *translation* | The translation vector (in pixels) |

#### 4.10.1.7 coordinate_system_update_dimensions()

```
void coordinate_system_update_dimensions (
            CoordinateSystem * cs,
            Vector2 position,
            Vector2 size )
```

Updates the dimensions of the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to update |
| *position* | The new position |
| *size* | The new size |

#### 4.10.1.8 coordinate_system_zoom()

```
void coordinate_system_zoom (
            CoordinateSystem * cs,
            double zoom )
```

Zooms into the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to zoom into |
| *zoom* | The zoom factor |

#### 4.10.1.9 coordinates_to_screen()

```
Vector2 coordinates_to_screen (
            CoordinateSystem * cs,
            Vector2 point )
```

Translates a point from the coordinate system to the screen.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to translate the point from |
| *point* | The point to translate |

**Returns**

Vector2 The translated point

#### 4.10.1.10 screen_to_coordinates()

```
Vector2 screen_to_coordinates (
            CoordinateSystem * cs,
            Vector2 point )
```

Translates a point from the screen to the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to translate the point to |
| *point* | The point to translate |

**Returns**

Vector2 The translated point

## 4.11 src/geometry/coordinate_system/coordinate_system.h File Reference

```
#include "../shape/shape.h"
#include "../vector2/vector2.h"
#include "../../texture/texture.h"
#include "../../utils/vector/vector.h"
```

### Classes

- struct CoordinateSystem

### Macros

- #define INITIAL_ZOOM 20

### Typedefs

- typedef struct CoordinateSystem CoordinateSystem

**Functions**

- CoordinateSystem ∗ coordinate_system_create (Vector2 position, Vector2 size, Vector2 origin)

  *Creates a coordinate system.*
- void coordinate_system_destroy (CoordinateSystem ∗cs)

  *Destroys a coordinate system.*
- Vector2 screen_to_coordinates (CoordinateSystem ∗cs, Vector2 point)

  *Translates a point from the screen to the coordinate system.*
- Vector2 coordinates_to_screen (CoordinateSystem ∗cs, Vector2 point)

  *Translates a point from the coordinate system to the screen.*
- bool coordinate_system_is_hovered (CoordinateSystem ∗cs, Vector2 point)

  *Returns whether the coordinate system is hovered by the point.*
- IShape ∗ coordinate_system_get_hovered_shape (CoordinateSystem ∗cs, Vector2 point)

  *Returns the shape hovered by the point.*
- void coordinate_system_translate (CoordinateSystem ∗cs, Vector2 translation)

  *Translates the coordinate system.*
- void coordinate_system_zoom (CoordinateSystem ∗cs, double zoom)

  *Zooms into the coordinate system.*
- void coordinate_system_update_dimensions (CoordinateSystem ∗cs, Vector2 position, Vector2 size)

  *Updates the dimensions of the coordinate system.*
- void coordinate_system_draw (CoordinateSystem ∗cs)

  *Draws the coordinate system.*

## 4.11.1 Macro Definition Documentation

### 4.11.1.1 INITIAL_ZOOM

```
#define INITIAL_ZOOM 20
```

## 4.11.2 Typedef Documentation

### 4.11.2.1 CoordinateSystem

```
typedef struct CoordinateSystem CoordinateSystem
```

## 4.11.3 Function Documentation

### 4.11.3.1 coordinate_system_create()

```
CoordinateSystem ∗ coordinate_system_create (
            Vector2 position,
            Vector2 size,
            Vector2 origin )
```

Creates a coordinate system.

**Parameters**

| | |
|---|---|
| *position* | The position of the coordinate system in the screen |
| *size* | The size of the coordinate system (in pixels) |
| *origin* | The origin of the coordinate system (relative to the coordinate system (normalized)) |

**Returns**

CoordinateSystem* The created coordinate system

### 4.11.3.2 coordinate_system_destroy()

```
void coordinate_system_destroy (
            CoordinateSystem * cs )
```

Destroys a coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to destroy |

### 4.11.3.3 coordinate_system_draw()

```
void coordinate_system_draw (
            CoordinateSystem * cs )
```

Draws the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to draw |

### 4.11.3.4 coordinate_system_get_hovered_shape()

```
IShape * coordinate_system_get_hovered_shape (
            CoordinateSystem * cs,
            Vector2 point )
```

Returns the shape hovered by the point.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to check |
| *point* | The point to check |

**Returns**

IShape∗ The hovered shape (NULL if none)

#### 4.11.3.5 coordinate_system_is_hovered()

```
bool coordinate_system_is_hovered (
            CoordinateSystem * cs,
            Vector2 point )
```

Returns whether the coordinate system is hovered by the point.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to check |
| *point* | The point to check |

#### 4.11.3.6 coordinate_system_translate()

```
void coordinate_system_translate (
            CoordinateSystem * cs,
            Vector2 translation )
```

Translates the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to translate |
| *translation* | The translation vector (in pixels) |

#### 4.11.3.7 coordinate_system_update_dimensions()

```
void coordinate_system_update_dimensions (
            CoordinateSystem * cs,
            Vector2 position,
            Vector2 size )
```

Updates the dimensions of the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to update |
| *position* | The new position |
| *size* | The new size |

#### 4.11.3.8 coordinate_system_zoom()

```
void coordinate_system_zoom (
            CoordinateSystem * cs,
            double zoom )
```

Zooms into the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to zoom into |
| *zoom* | The zoom factor |

#### 4.11.3.9 coordinates_to_screen()

```
Vector2 coordinates_to_screen (
            CoordinateSystem * cs,
            Vector2 point )
```

Translates a point from the coordinate system to the screen.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to translate the point from |
| *point* | The point to translate |

**Returns**

Vector2 The translated point

#### 4.11.3.10 screen_to_coordinates()

```
Vector2 screen_to_coordinates (
            CoordinateSystem * cs,
            Vector2 point )
```

Translates a point from the screen to the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to translate the point to |
| *point* | The point to translate |

**Returns**

Vector2 The translated point

## 4.12 coordinate_system.h

Go to the documentation of this file.

```
#pragma once

#include "../shape/shape.h"
#include "../vector2/vector2.h"
#include "../../texture/texture.h"
#include "../../utils/vector/vector.h"

#define INITIAL_ZOOM 20

typedef struct CoordinateSystem
{
    Vector2 position;
    Vector2 size;
    Vector2 origin;
    double zoom;

    Vector* shapes;
} CoordinateSystem;

CoordinateSystem* coordinate_system_create(Vector2 position, Vector2 size, Vector2 origin);
void coordinate_system_destroy(CoordinateSystem* cs);

Vector2 screen_to_coordinates(CoordinateSystem* cs, Vector2 point);
Vector2 coordinates_to_screen(CoordinateSystem* cs, Vector2 point);

bool coordinate_system_is_hovered(CoordinateSystem* cs, Vector2 point);
IShape* coordinate_system_get_hovered_shape(CoordinateSystem* cs, Vector2 point);

void coordinate_system_translate(CoordinateSystem* cs, Vector2 translation);
void coordinate_system_zoom(CoordinateSystem* cs, double zoom);
void coordinate_system_update_dimensions(CoordinateSystem* cs, Vector2 position, Vector2 size);
void coordinate_system_draw(CoordinateSystem* cs);
```

## 4.13 src/geometry/shape/shape.c File Reference

```
#include "shape.h"
#include "../coordinate_system/coordinate_system.h"
#include "../../renderer/renderer.h"
#include <math.h>
```

### Functions

- Point ∗ point_create (CoordinateSystem ∗cs, Vector2 coordinates)

  *Creates a point in the coordinate system.*
- Line ∗ line_create (CoordinateSystem ∗cs, Point ∗p1, Point ∗p2)

  *Creates a line in the coordinate system.*
- Circle ∗ circle_create (CoordinateSystem ∗cs, Point ∗center, Point ∗perimeter_point)

  *Creates a circle in the coordinate system.*

## 4.13.1 Function Documentation

### 4.13.1.1 circle_create()

```
Circle * circle_create (
             CoordinateSystem * cs,
             Point * center,
             Point * perimeter_point )
```

Creates a circle in the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to create the circle in |
| *center* | The center of the circle |
| *perimeter_point* | A point on the perimeter of the circle (has to be different from center) |

**Returns**

Circle∗ The created circle

### 4.13.1.2 line_create()

```
Line * line_create (
             CoordinateSystem * cs,
             Point * p1,
             Point * p2 )
```

Creates a line in the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to create the line in |
| *p1* | A point of the line |
| *p2* | Another point of the line (has to be different from p1) |

**Returns**

Line∗ The created line

### 4.13.1.3 point_create()

```
Point * point_create (
             CoordinateSystem * cs,
             Vector2 coordinates )
```

Creates a point in the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to create the point in |
| *coordinates* | The coordinates of the point |

**Returns**

Point∗ The created point

## 4.14 src/geometry/shape/shape.h File Reference

```
#include <stdbool.h>
#include "../vector2/vector2.h"
```

### Classes

- struct IShape
- struct Point
- struct Line
- struct Circle

### Macros

- #define OVERLAP_DISTANCE 5

### Typedefs

- typedef struct CoordinateSystem CoordinateSystem
- typedef struct IShape IShape
- typedef struct Point Point
- typedef struct Line Line
- typedef struct Circle Circle

### Functions

- Point ∗ point_create (CoordinateSystem ∗cs, Vector2 coordinates)

  *Creates a point in the coordinate system.*
- Line ∗ line_create (CoordinateSystem ∗cs, Point ∗p1, Point ∗p2)

  *Creates a line in the coordinate system.*
- Circle ∗ circle_create (CoordinateSystem ∗cs, Point ∗center, Point ∗perimeter_point)

  *Creates a circle in the coordinate system.*

### 4.14.1 Macro Definition Documentation

#### 4.14.1.1 OVERLAP_DISTANCE

```
#define OVERLAP_DISTANCE 5
```

## 4.14.2 Typedef Documentation

### 4.14.2.1 Circle

```
typedef struct Circle Circle
```

### 4.14.2.2 CoordinateSystem

```
typedef struct CoordinateSystem CoordinateSystem
```

### 4.14.2.3 IShape

```
typedef struct IShape IShape
```

### 4.14.2.4 Line

```
typedef struct Line Line
```

### 4.14.2.5 Point

```
typedef struct Point Point
```

## 4.14.3 Function Documentation

### 4.14.3.1 circle_create()

```
Circle * circle_create (
            CoordinateSystem * cs,
            Point * center,
            Point * perimeter_point )
```

Creates a circle in the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to create the circle in |
| *center* | The center of the circle |
| *perimeter_point* | A point on the perimeter of the circle (has to be different from center) |

**Returns**

Circle* The created circle

#### 4.14.3.2 line_create()

```
Line * line_create (
             CoordinateSystem * cs,
             Point * p1,
             Point * p2 )
```

Creates a line in the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to create the line in |
| *p1* | A point of the line |
| *p2* | Another point of the line (has to be different from p1) |

**Returns**

Line* The created line

#### 4.14.3.3 point_create()

```
Point * point_create (
             CoordinateSystem * cs,
             Vector2 coordinates )
```

Creates a point in the coordinate system.

**Parameters**

| | |
|---|---|
| *cs* | The coordinate system to create the point in |
| *coordinates* | The coordinates of the point |

**Returns**

Point* The created point

## 4.15 shape.h

Go to the documentation of this file.

```
#pragma once

#include <stdbool.h>

#include "../vector2/vector2.h"

#define OVERLAP_DISTANCE 5

typedef struct CoordinateSystem CoordinateSystem;
typedef struct IShape IShape;
typedef struct IShape
{
    void (*draw)(CoordinateSystem* cs, IShape* self);
    void (*translate)(CoordinateSystem* cs, IShape* self, Vector2 translation);
    void (*destroy)(CoordinateSystem* cs, IShape* self);
    bool (*overlap_point)(CoordinateSystem* cs, IShape* self, Vector2 point);
    bool (*is_defined_by)(IShape* self, IShape* shape);
} IShape;

typedef struct Point
{
    IShape base;
    Vector2 coordinates;
} Point;

typedef struct Line
{
    IShape base;
    Point *p1, *p2;
} Line;

typedef struct Circle
{
    IShape base;
    Point* center;
    Point* perimeter_point;
} Circle;

Point* point_create(CoordinateSystem* cs, Vector2 coordinates);
Line* line_create(CoordinateSystem* cs, Point* p1, Point* p2);
Circle* circle_create(CoordinateSystem* cs, Point* center, Point* perimeter_point);
```

## 4.16 src/geometry/vector2/vector2.c File Reference

```
#include "vector2.h"
#include <math.h>
```

### Functions

- Vector2 vector2_create (double x, double y)
- Vector2 vector2_from_polar (double angle, double length)
- Vector2 vector2_from_point (SDL_Point point)
- Vector2 vector2_zero ()
- Vector2 vector2_one ()
- Vector2 vector2_up ()
- Vector2 vector2_down ()
- Vector2 vector2_left ()
- Vector2 vector2_right ()
- Vector2 vector2_add (Vector2 a, Vector2 b)
- Vector2 vector2_subtract (Vector2 a, Vector2 b)
- Vector2 vector2_scale (Vector2 a, double b)
- Vector2 vector2_negate (Vector2 a)
- Vector2 vector2_multiply (Vector2 a, Vector2 b)
- Vector2 vector2_divide (Vector2 a, Vector2 b)
- double vector2_dot (Vector2 a, Vector2 b)
- double vector2_length (Vector2 a)
- double vector2_distance (Vector2 a, Vector2 b)
- double vector2_angle (Vector2 a)
- Vector2 vector2_normalize (Vector2 a)
- Vector2 vector2_rotate90 (Vector2 a)
- Vector2 vector2_rotate (Vector2 a, double angle)
- Vector2 vector2_reflect (Vector2 a, Vector2 normal)

## 4.16.1 Function Documentation

### 4.16.1.1 vector2_add()

```
Vector2 vector2_add (
            Vector2 a,
            Vector2 b )
```

### 4.16.1.2 vector2_angle()

```
double vector2_angle (
            Vector2 a )
```

### 4.16.1.3 vector2_create()

```
Vector2 vector2_create (
            double x,
            double y )
```

### 4.16.1.4 vector2_distance()

```
double vector2_distance (
            Vector2 a,
            Vector2 b )
```

### 4.16.1.5 vector2_divide()

```
Vector2 vector2_divide (
            Vector2 a,
            Vector2 b )
```

### 4.16.1.6 vector2_dot()

```
double vector2_dot (
            Vector2 a,
            Vector2 b )
```

### 4.16.1.7 vector2_down()

```
Vector2 vector2_down ( )
```

### 4.16.1.8 vector2_from_point()

```
Vector2 vector2_from_point (
            SDL_Point point )
```

#### 4.16.1.9 vector2_from_polar()

```
Vector2 vector2_from_polar (
            double angle,
            double length )
```

#### 4.16.1.10 vector2_left()

```
Vector2 vector2_left ( )
```

#### 4.16.1.11 vector2_length()

```
double vector2_length (
            Vector2 a )
```

#### 4.16.1.12 vector2_multiply()

```
Vector2 vector2_multiply (
            Vector2 a,
            Vector2 b )
```

#### 4.16.1.13 vector2_negate()

```
Vector2 vector2_negate (
            Vector2 a )
```

#### 4.16.1.14 vector2_normalize()

```
Vector2 vector2_normalize (
            Vector2 a )
```

#### 4.16.1.15 vector2_one()

```
Vector2 vector2_one ( )
```

#### 4.16.1.16 vector2_reflect()

```
Vector2 vector2_reflect (
            Vector2 a,
            Vector2 normal )
```

#### 4.16.1.17 vector2_right()

```
Vector2 vector2_right ( )
```

#### 4.16.1.18 vector2_rotate()

```
Vector2 vector2_rotate (
            Vector2 a,
            double angle )
```

#### 4.16.1.19 vector2_rotate90()

```
Vector2 vector2_rotate90 (
            Vector2 a )
```

#### 4.16.1.20 vector2_scale()

```
Vector2 vector2_scale (
            Vector2 a,
            double b )
```

#### 4.16.1.21 vector2_subtract()

```
Vector2 vector2_subtract (
            Vector2 a,
            Vector2 b )
```

#### 4.16.1.22 vector2_up()

```
Vector2 vector2_up ( )
```

#### 4.16.1.23 vector2_zero()

```
Vector2 vector2_zero ( )
```

## 4.17 src/geometry/vector2/vector2.h File Reference

### Classes

- struct Vector2

### Typedefs

- typedef struct Vector2 Vector2

### Functions

- Vector2 vector2_create (double x, double y)
- Vector2 vector2_from_polar (double angle, double length)
- Vector2 vector2_from_point (SDL_Point point)
- Vector2 vector2_zero ()
- Vector2 vector2_one ()
- Vector2 vector2_up ()
- Vector2 vector2_down ()
- Vector2 vector2_left ()
- Vector2 vector2_right ()
- Vector2 vector2_add (Vector2 a, Vector2 b)
- Vector2 vector2_subtract (Vector2 a, Vector2 b)
- Vector2 vector2_scale (Vector2 a, double b)
- Vector2 vector2_negate (Vector2 a)
- Vector2 vector2_multiply (Vector2 a, Vector2 b)
- Vector2 vector2_divide (Vector2 a, Vector2 b)
- double vector2_dot (Vector2 a, Vector2 b)
- double vector2_length (Vector2 a)
- double vector2_distance (Vector2 a, Vector2 b)
- double vector2_angle (Vector2 a)
- Vector2 vector2_normalize (Vector2 a)
- Vector2 vector2_rotate90 (Vector2 a)
- Vector2 vector2_rotate (Vector2 a, double angle)
- Vector2 vector2_reflect (Vector2 a, Vector2 normal)

## 4.17.1 Typedef Documentation

### 4.17.1.1 Vector2

```
typedef struct Vector2 Vector2
```

## 4.17.2 Function Documentation

### 4.17.2.1 vector2_add()

```
Vector2 vector2_add (
            Vector2 a,
            Vector2 b )
```

### 4.17.2.2 vector2_angle()

```
double vector2_angle (
            Vector2 a )
```

### 4.17.2.3 vector2_create()

```
Vector2 vector2_create (
            double x,
            double y )
```

#### 4.17.2.4 vector2_distance()

```
double vector2_distance (
            Vector2 a,
            Vector2 b )
```

#### 4.17.2.5 vector2_divide()

```
Vector2 vector2_divide (
            Vector2 a,
            Vector2 b )
```

#### 4.17.2.6 vector2_dot()

```
double vector2_dot (
            Vector2 a,
            Vector2 b )
```

#### 4.17.2.7 vector2_down()

```
Vector2 vector2_down ( )
```

#### 4.17.2.8 vector2_from_point()

```
Vector2 vector2_from_point (
            SDL_Point point )
```

#### 4.17.2.9 vector2_from_polar()

```
Vector2 vector2_from_polar (
            double angle,
            double length )
```

#### 4.17.2.10 vector2_left()

```
Vector2 vector2_left ( )
```

#### 4.17.2.11 vector2_length()

```
double vector2_length (
            Vector2 a )
```

#### 4.17.2.12 vector2_multiply()

```
Vector2 vector2_multiply (
            Vector2 a,
            Vector2 b )
```

#### 4.17.2.13 vector2_negate()

```
Vector2 vector2_negate (
            Vector2 a )
```

#### 4.17.2.14 vector2_normalize()

```
Vector2 vector2_normalize (
            Vector2 a )
```

#### 4.17.2.15 vector2_one()

```
Vector2 vector2_one ( )
```

#### 4.17.2.16 vector2_reflect()

```
Vector2 vector2_reflect (
            Vector2 a,
            Vector2 normal )
```

#### 4.17.2.17 vector2_right()

```
Vector2 vector2_right ( )
```

#### 4.17.2.18 vector2_rotate()

```
Vector2 vector2_rotate (
            Vector2 a,
            double angle )
```

#### 4.17.2.19 vector2_rotate90()

```
Vector2 vector2_rotate90 (
            Vector2 a )
```

#### 4.17.2.20 vector2_scale()

```
Vector2 vector2_scale (
              Vector2 a,
              double b )
```

#### 4.17.2.21 vector2_subtract()

```
Vector2 vector2_subtract (
              Vector2 a,
              Vector2 b )
```

#### 4.17.2.22 vector2_up()

```
Vector2 vector2_up ( )
```

#### 4.17.2.23 vector2_zero()

```
Vector2 vector2_zero ( )
```

## 4.18 vector2.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
#endif

typedef struct Vector2
{
    double x, y;
} Vector2;

Vector2 vector2_create(double x, double y);
Vector2 vector2_from_polar(double angle, double length);
Vector2 vector2_from_point(SDL_Point point);

Vector2 vector2_zero();
Vector2 vector2_one();
Vector2 vector2_up();
Vector2 vector2_down();
Vector2 vector2_left();
Vector2 vector2_right();

Vector2 vector2_add(Vector2 a, Vector2 b);
Vector2 vector2_subtract(Vector2 a, Vector2 b);
Vector2 vector2_scale(Vector2 a, double b);
Vector2 vector2_negate(Vector2 a);
Vector2 vector2_multiply(Vector2 a, Vector2 b);
Vector2 vector2_divide(Vector2 a, Vector2 b);
double vector2_dot(Vector2 a, Vector2 b);
double vector2_length(Vector2 a);
double vector2_distance(Vector2 a, Vector2 b);
double vector2_angle(Vector2 a);
Vector2 vector2_normalize(Vector2 a);
Vector2 vector2_rotate90(Vector2 a);
Vector2 vector2_rotate(Vector2 a, double angle);
Vector2 vector2_reflect(Vector2 a, Vector2 normal);
```

## 4.19 src/includes.h File Reference

```
#include "app/app.h"
#include "color/color.h"
#include "font/font.h"
#include "geometry/coordinate_system/coordinate_system.h"
#include "geometry/shape/shape.h"
#include "geometry/vector2/vector2.h"
#include "input/input.h"
#include "renderer/renderer.h"
#include "texture/texture.h"
#include "ui/ui.h"
#include "ui/ui_constraint/ui_constraint.h"
#include "ui/ui_element/ui_element.h"
#include "utils/math/math.h"
#include "utils/vector/vector.h"
#include "window/window.h"
```

## 4.20 includes.h

Go to the documentation of this file.

```
#pragma once

#include "app/app.h"
#include "color/color.h"
#include "font/font.h"
#include "geometry/coordinate_system/coordinate_system.h"
#include "geometry/shape/shape.h"
#include "geometry/vector2/vector2.h"
#include "input/input.h"
#include "renderer/renderer.h"
#include "texture/texture.h"
#include "ui/ui.h"
#include "ui/ui_constraint/ui_constraint.h"
#include "ui/ui_element/ui_element.h"
#include "utils/math/math.h"
#include "utils/vector/vector.h"
#include "window/window.h"
```

## 4.21 src/input/input.c File Reference

```
#include "input.h"
#include "../app/app.h"
```

### Functions

- bool input_is_mouse_button_down (int button)
- bool input_is_mouse_button_pressed (int button)
- bool input_is_mouse_button_released (int button)
- bool input_is_key_down (int key)
- bool input_is_key_pressed (int key)
- bool input_is_key_released (int key)
- SDL_Point input_get_mouse_position ()
- SDL_Point input_get_mouse_motion ()

- int input_get_mouse_wheel_delta ()
- void _input_init (InputData ∗input_data)
- void _input_handle_event (InputData ∗input_data, SDL_Event ∗event)
- void _input_reset (InputData ∗input_data)
- void _input_close (InputData ∗input_data)
- void _input_set_target (InputData ∗input_data)

**Variables**

- InputData ∗ target_input_data

## 4.21.1 Function Documentation

### 4.21.1.1 _input_close()

```
void _input_close (
            InputData * input_data )
```

### 4.21.1.2 _input_handle_event()

```
void _input_handle_event (
            InputData * input_data,
            SDL_Event * event )
```

### 4.21.1.3 _input_init()

```
void _input_init (
            InputData * input_data )
```

### 4.21.1.4 _input_reset()

```
void _input_reset (
            InputData * input_data )
```

### 4.21.1.5 _input_set_target()

```
void _input_set_target (
            InputData * input_data )
```

### 4.21.1.6 input_get_mouse_motion()

```
SDL_Point input_get_mouse_motion ( )
```

#### 4.21.1.7 input_get_mouse_position()

```
SDL_Point input_get_mouse_position ( )
```

#### 4.21.1.8 input_get_mouse_wheel_delta()

```
int input_get_mouse_wheel_delta ( )
```

#### 4.21.1.9 input_is_key_down()

```
bool input_is_key_down (
             int key )
```

#### 4.21.1.10 input_is_key_pressed()

```
bool input_is_key_pressed (
             int key )
```

#### 4.21.1.11 input_is_key_released()

```
bool input_is_key_released (
             int key )
```

#### 4.21.1.12 input_is_mouse_button_down()

```
bool input_is_mouse_button_down (
             int button )
```

#### 4.21.1.13 input_is_mouse_button_pressed()

```
bool input_is_mouse_button_pressed (
             int button )
```

#### 4.21.1.14 input_is_mouse_button_released()

```
bool input_is_mouse_button_released (
             int button )
```

### 4.21.2 Variable Documentation

#### 4.21.2.1 target_input_data

```
InputData* target_input_data
```

## 4.22 src/input/input.h File Reference

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

### Classes

- struct InputData

### Typedefs

- typedef struct InputData InputData

### Functions

- bool input_is_mouse_button_down (int button)
- bool input_is_mouse_button_pressed (int button)
- bool input_is_mouse_button_released (int button)
- bool input_is_key_down (int key)
- bool input_is_key_pressed (int key)
- bool input_is_key_released (int key)
- SDL_Point input_get_mouse_position ()
- SDL_Point input_get_mouse_motion ()
- int input_get_mouse_wheel_delta ()
- void _input_init (InputData ∗input_data)
- void _input_handle_event (InputData ∗input_data, SDL_Event ∗event)
- void _input_reset (InputData ∗input_data)
- void _input_close (InputData ∗input_data)
- void _input_set_target (InputData ∗input_data)

### 4.22.1 Typedef Documentation

#### 4.22.1.1 InputData

```
typedef struct InputData InputData
```

### 4.22.2 Function Documentation

#### 4.22.2.1 _input_close()

```
void _input_close (
            InputData * input_data )
```

#### 4.22.2.2 _input_handle_event()

```
void _input_handle_event (
            InputData * input_data,
            SDL_Event * event )
```

#### 4.22.2.3 _input_init()

```
void _input_init (
            InputData * input_data )
```

#### 4.22.2.4 _input_reset()

```
void _input_reset (
            InputData * input_data )
```

#### 4.22.2.5 _input_set_target()

```
void _input_set_target (
            InputData * input_data )
```

#### 4.22.2.6 input_get_mouse_motion()

```
SDL_Point input_get_mouse_motion ( )
```

#### 4.22.2.7 input_get_mouse_position()

```
SDL_Point input_get_mouse_position ( )
```

#### 4.22.2.8 input_get_mouse_wheel_delta()

```
int input_get_mouse_wheel_delta ( )
```

#### 4.22.2.9 input_is_key_down()

```
bool input_is_key_down (
            int key )
```

#### 4.22.2.10 input_is_key_pressed()

```
bool input_is_key_pressed (
            int key )
```

#### 4.22.2.11 input_is_key_released()

```
bool input_is_key_released (
            int key )
```

#### 4.22.2.12 input_is_mouse_button_down()

```
bool input_is_mouse_button_down (
              int button )
```

#### 4.22.2.13 input_is_mouse_button_pressed()

```
bool input_is_mouse_button_pressed (
              int button )
```

#### 4.22.2.14 input_is_mouse_button_released()

```
bool input_is_mouse_button_released (
              int button )
```

## 4.23 input.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
#endif

#include <stdbool.h>

typedef struct InputData
{
    //mouse
    bool current_mouse_button_state[5];
    bool old_mouse_button_state[5];
    SDL_Point current_mouse_position;
    SDL_Point old_mouse_position;
    int mouse_wheel_delta;

    //keyboard
    Uint8* current_keyboard_state;
    Uint8* old_keyboard_state;
    int key_count;
} InputData;

//API functions
bool input_is_mouse_button_down(int button);
bool input_is_mouse_button_pressed(int button);
bool input_is_mouse_button_released(int button);

bool input_is_key_down(int key);
bool input_is_key_pressed(int key);
bool input_is_key_released(int key);

SDL_Point input_get_mouse_position();
SDL_Point input_get_mouse_motion();
int input_get_mouse_wheel_delta();

//internal functions
void _input_init(InputData* input_data);
void _input_handle_event(InputData* input_data, SDL_Event* event);
void _input_reset(InputData* input_data);
void _input_close(InputData* input_data);
void _input_set_target(InputData* input_data);
```

## 4.24 src/main.c File Reference

This is the entry point of the application.

```
#include "includes.h"
#include "utils/vector/vector.h"
```

### Macros

- #define FPS 60
- #define MOUSE_WHEEL_SENSITIVITY 5

### Typedefs

- typedef enum State State

### Enumerations

- enum State {
  STATE_POINTER , STATE_CS_DRAGGED , STATE_POINT , STATE_LINE ,
  STATE_LINE_POINT1_PLACED , STATE_CIRCLE , STATE_CIRCLE_CENTER_PLACED }

### Functions

- void on_pointer_clicked (UIButton ∗self)
- void on_point_clicked (UIButton ∗self)
- void on_line_clicked (UIButton ∗self)
- void on_circle_clicked (UIButton ∗self)
- void on_filemenu_clicked (UISplitButton ∗self, Sint32 index)
- void on_editmenu_clicked (UISplitButton ∗self, Sint32 index)
- void on_canvas_size_changed (UIContainer ∗self, SDL_Point size)
- int main (void)
- void on_pointer_clicked (UIButton ∗self __attribute__((unused)))
- void on_point_clicked (UIButton ∗self __attribute__((unused)))
- void on_line_clicked (UIButton ∗self __attribute__((unused)))
- void on_circle_clicked (UIButton ∗self __attribute__((unused)))
- void on_filemenu_clicked (UISplitButton ∗self __attribute__((unused)), Sint32 index __attribute__((unused)))
- void on_editmenu_clicked (UISplitButton ∗self __attribute__((unused)), Sint32 index __attribute__((unused)))

### Variables

- CoordinateSystem ∗ cs
- State state = STATE_POINTER

## 4.24.1 Detailed Description

This is the entry point of the application.

**Author**

Örs Mándli ( mandliors@gmail.com)

**Version**

0.1

**Date**

2023-11-05

**Copyright**

Copyright (c) 2023

## 4.24.2 Macro Definition Documentation

### 4.24.2.1 FPS

```
#define FPS 60
```

### 4.24.2.2 MOUSE_WHEEL_SENSITIVITY

```
#define MOUSE_WHEEL_SENSITIVITY 5
```

## 4.24.3 Typedef Documentation

### 4.24.3.1 State

```
typedef enum State State
```

## 4.24.4 Enumeration Type Documentation

### 4.24.4.1 State

```
enum State
```

**Enumerator**

| | |
|---|---|
| STATE_POINTER | |
| STATE_CS_DRAGGED | |
| STATE_POINT | |
| STATE_LINE | |
| STATE_LINE_POINT1_PLACED | |
| STATE_CIRCLE | |

## 4.24.5 Function Documentation

### 4.24.5.1 main()

```
int main (
            void  )
```

### 4.24.5.2 on_canvas_size_changed()

```
void on_canvas_size_changed (
            UIContainer * self,
            SDL_Point size )
```

### 4.24.5.3 on_circle_clicked() [1/2]

```
void on_circle_clicked (
            UIButton *self  __attribute__(unused) )
```

### 4.24.5.4 on_circle_clicked() [2/2]

```
void on_circle_clicked (
            UIButton * self )
```

### 4.24.5.5 on_editmenu_clicked() [1/2]

```
void on_editmenu_clicked (
            UISplitButton *self  __attribute__(unused),
            Sint32 index  __attribute__(unused) )
```

### 4.24.5.6 on_editmenu_clicked() [2/2]

```
void on_editmenu_clicked (
            UISplitButton * self,
            Sint32 index )
```

### 4.24.5.7 on_filemenu_clicked() [1/2]

```
void on_filemenu_clicked (
            UISplitButton *self  __attribute__(unused),
            Sint32 index  __attribute__(unused) )
```

### 4.24.5.8 on_filemenu_clicked() [2/2]

```
void on_filemenu_clicked (
            UISplitButton * self,
            Sint32 index )
```

#### 4.24.5.9 on_line_clicked() [1/2]

```
void on_line_clicked (
            UIButton *self  __attribute__(unused) )
```

#### 4.24.5.10 on_line_clicked() [2/2]

```
void on_line_clicked (
            UIButton * self )
```

#### 4.24.5.11 on_point_clicked() [1/2]

```
void on_point_clicked (
            UIButton *self  __attribute__(unused) )
```

#### 4.24.5.12 on_point_clicked() [2/2]

```
void on_point_clicked (
            UIButton * self )
```

#### 4.24.5.13 on_pointer_clicked() [1/2]

```
void on_pointer_clicked (
            UIButton *self  __attribute__(unused) )
```

#### 4.24.5.14 on_pointer_clicked() [2/2]

```
void on_pointer_clicked (
            UIButton * self )
```

### 4.24.6 Variable Documentation

#### 4.24.6.1 cs

```
CoordinateSystem* cs
```

#### 4.24.6.2 state

```
State state = STATE_POINTER
```

## 4.25 src/renderer/renderer.c File Reference

```
#include "renderer.h"
#include "../texture/texture.h"
#include "../font/font.h"
```

**Functions**

- void renderer_set_default_font (Font ∗font)
- void renderer_set_clip_rect (int x, int y, int width, int height)
- void renderer_reset_clip_rect ()
- Texture ∗ renderer_create_framebuffer (int width, int height)
- void renderer_resize_framebuffer (Texture ∗framebuffer, int width, int height)
- void renderer_bind_framebuffer (Texture ∗framebuffer)
- void renderer_clear (Color color)
- void renderer_draw_pixel (int x, int y, Color color)
- void renderer_draw_line (int x1, int y1, int x2, int y2, int thickness, Color color)
- void renderer_draw_rect (int x, int y, int width, int height, Color color)
- void renderer_draw_filled_rect (int x, int y, int width, int height, Color color)
- void renderer_draw_circle (int x, int y, int radius, Color color)
- void renderer_draw_filled_circle (int x, int y, int radius, Color color)
- void renderer_draw_ellipse (int x, int y, int rx, int ry, Color color)
- void renderer_draw_filled_ellipse (int x, int y, int rx, int ry, Color color)
- void renderer_draw_triangle (int x1, int y1, int x2, int y2, int x3, int y3, Color color)
- void renderer_draw_filled_triangle (int x1, int y1, int x2, int y2, int x3, int y3, Color color)
- void renderer_draw_rounded_rect (int x, int y, int width, int height, int radius, Color color)
- void renderer_draw_filled_rounded_rect (int x, int y, int width, int height, int radius, Color color)
- void renderer_draw_polygon (const short ∗vx, const short ∗vy, int n, Color color)
- void renderer_draw_filled_polygon (const short ∗vx, const short ∗vy, int n, Color color)
- void renderer_draw_arc (int x, int y, int radius, int start, int end, Color color)
- void renderer_draw_pie (int x, int y, int radius, int start, int end, Color color)
- void renderer_draw_filled_pie (int x, int y, int radius, int start, int end, Color color)
- void renderer_draw_bezier (const short ∗vx, const short ∗vy, int n, int s, Color color)
- void renderer_draw_texture (Texture ∗texture, int x, int y, int width, int height)
- void renderer_draw_text (const char ∗text, int x, int y, Color color)
- SDL_Point renderer_query_text_size (const char ∗text)
- void _renderer_set_target (SDL_Renderer ∗renderer)

**Variables**

- SDL_Renderer ∗ target_renderer
- Font ∗ default_font

### 4.25.1 Function Documentation

#### 4.25.1.1 _renderer_set_target()

```
void _renderer_set_target (
          SDL_Renderer ∗ renderer )
```

#### 4.25.1.2 renderer_bind_framebuffer()

```
void renderer_bind_framebuffer (
             Texture * framebuffer )
```

#### 4.25.1.3 renderer_clear()

```
void renderer_clear (
             Color color )
```

#### 4.25.1.4 renderer_create_framebuffer()

```
Texture * renderer_create_framebuffer (
             int width,
             int height )
```

#### 4.25.1.5 renderer_draw_arc()

```
void renderer_draw_arc (
             int x,
             int y,
             int radius,
             int start,
             int end,
             Color color )
```

#### 4.25.1.6 renderer_draw_bezier()

```
void renderer_draw_bezier (
             const short * vx,
             const short * vy,
             int n,
             int s,
             Color color )
```

#### 4.25.1.7 renderer_draw_circle()

```
void renderer_draw_circle (
             int x,
             int y,
             int radius,
             Color color )
```

#### 4.25.1.8 renderer_draw_ellipse()

```
void renderer_draw_ellipse (
             int x,
             int y,
             int rx,
             int ry,
             Color color )
```

#### 4.25.1.9 renderer_draw_filled_circle()

```
void renderer_draw_filled_circle (
             int x,
             int y,
             int radius,
             Color color )
```

#### 4.25.1.10 renderer_draw_filled_ellipse()

```
void renderer_draw_filled_ellipse (
             int x,
             int y,
             int rx,
             int ry,
             Color color )
```

#### 4.25.1.11 renderer_draw_filled_pie()

```
void renderer_draw_filled_pie (
             int x,
             int y,
             int radius,
             int start,
             int end,
             Color color )
```

#### 4.25.1.12 renderer_draw_filled_polygon()

```
void renderer_draw_filled_polygon (
             const short * vx,
             const short * vy,
             int n,
             Color color )
```

#### 4.25.1.13 renderer_draw_filled_rect()

```
void renderer_draw_filled_rect (
             int x,
             int y,
             int width,
             int height,
             Color color )
```

#### 4.25.1.14 renderer_draw_filled_rounded_rect()

```
void renderer_draw_filled_rounded_rect (
             int x,
             int y,
             int width,
             int height,
             int radius,
             Color color )
```

#### 4.25.1.15 renderer_draw_filled_triangle()

```
void renderer_draw_filled_triangle (
            int x1,
            int y1,
            int x2,
            int y2,
            int x3,
            int y3,
            Color color )
```

#### 4.25.1.16 renderer_draw_line()

```
void renderer_draw_line (
            int x1,
            int y1,
            int x2,
            int y2,
            int thickness,
            Color color )
```

#### 4.25.1.17 renderer_draw_pie()

```
void renderer_draw_pie (
            int x,
            int y,
            int radius,
            int start,
            int end,
            Color color )
```

#### 4.25.1.18 renderer_draw_pixel()

```
void renderer_draw_pixel (
            int x,
            int y,
            Color color )
```

#### 4.25.1.19 renderer_draw_polygon()

```
void renderer_draw_polygon (
            const short * vx,
            const short * vy,
            int n,
            Color color )
```

#### 4.25.1.20 renderer_draw_rect()

```
void renderer_draw_rect (
             int x,
             int y,
             int width,
             int height,
             Color color )
```

#### 4.25.1.21 renderer_draw_rounded_rect()

```
void renderer_draw_rounded_rect (
             int x,
             int y,
             int width,
             int height,
             int radius,
             Color color )
```

#### 4.25.1.22 renderer_draw_text()

```
void renderer_draw_text (
             const char * text,
             int x,
             int y,
             Color color )
```

#### 4.25.1.23 renderer_draw_texture()

```
void renderer_draw_texture (
             Texture * texture,
             int x,
             int y,
             int width,
             int height )
```

#### 4.25.1.24 renderer_draw_triangle()

```
void renderer_draw_triangle (
             int x1,
             int y1,
             int x2,
             int y2,
             int x3,
             int y3,
             Color color )
```

#### 4.25.1.25 renderer_query_text_size()

```
SDL_Point renderer_query_text_size (
             const char * text )
```

#### 4.25.1.26 renderer_reset_clip_rect()

```
void renderer_reset_clip_rect ( )
```

#### 4.25.1.27 renderer_resize_framebuffer()

```
void renderer_resize_framebuffer (
            Texture * framebuffer,
            int width,
            int height )
```

#### 4.25.1.28 renderer_set_clip_rect()

```
void renderer_set_clip_rect (
            int x,
            int y,
            int width,
            int height )
```

#### 4.25.1.29 renderer_set_default_font()

```
void renderer_set_default_font (
            Font * font )
```

### 4.25.2 Variable Documentation

#### 4.25.2.1 default_font

```
Font* default_font
```

#### 4.25.2.2 target_renderer

```
SDL_Renderer* target_renderer
```

## 4.26 src/renderer/renderer.h File Reference

```
#include "../color/color.h"
#include "../font/font.h"
#include "../texture/texture.h"
```

### Functions

- void renderer_set_default_font (Font ∗font)
- void renderer_set_clip_rect (int x, int y, int width, int height)
- void renderer_reset_clip_rect ()
- Texture ∗ renderer_create_framebuffer (int width, int height)
- void renderer_resize_framebuffer (Texture ∗framebuffer, int width, int height)
- void renderer_bind_framebuffer (Texture ∗framebuffer)
- void renderer_clear (Color color)
- void renderer_draw_pixel (int x, int y, Color color)
- void renderer_draw_line (int x1, int y1, int x2, int y2, int thickness, Color color)
- void renderer_draw_rect (int x, int y, int width, int height, Color color)
- void renderer_draw_filled_rect (int x, int y, int width, int height, Color color)
- void renderer_draw_circle (int x, int y, int radius, Color color)
- void renderer_draw_filled_circle (int x, int y, int radius, Color color)
- void renderer_draw_ellipse (int x, int y, int rx, int ry, Color color)
- void renderer_draw_filled_ellipse (int x, int y, int rx, int ry, Color color)
- void renderer_draw_triangle (int x1, int y1, int x2, int y2, int x3, int y3, Color color)
- void renderer_draw_filled_triangle (int x1, int y1, int x2, int y2, int x3, int y3, Color color)
- void renderer_draw_rounded_rect (int x, int y, int width, int height, int radius, Color color)
- void renderer_draw_filled_rounded_rect (int x, int y, int width, int height, int radius, Color color)
- void renderer_draw_polygon (const short ∗vx, const short ∗vy, int n, Color color)
- void renderer_draw_filled_polygon (const short ∗vx, const short ∗vy, int n, Color color)
- void renderer_draw_arc (int x, int y, int radius, int start, int end, Color color)
- void renderer_draw_pie (int x, int y, int radius, int start, int end, Color color)
- void renderer_draw_filled_pie (int x, int y, int radius, int start, int end, Color color)
- void renderer_draw_bezier (const short ∗vx, const short ∗vy, int n, int s, Color color)
- void renderer_draw_texture (Texture ∗texture, int x, int y, int width, int height)
- void renderer_draw_text (const char ∗text, int x, int y, Color color)
- SDL_Point renderer_query_text_size (const char ∗text)
- void _renderer_set_target (SDL_Renderer ∗renderer)

## 4.26.1 Function Documentation

### 4.26.1.1 _renderer_set_target()

```
void _renderer_set_target (
            SDL_Renderer * renderer )
```

### 4.26.1.2 renderer_bind_framebuffer()

```
void renderer_bind_framebuffer (
            Texture * framebuffer )
```

### 4.26.1.3 renderer_clear()

```
void renderer_clear (
            Color color )
```

### 4.26.1.4 renderer_create_framebuffer()

```
Texture * renderer_create_framebuffer (
             int width,
             int height )
```

### 4.26.1.5 renderer_draw_arc()

```
void renderer_draw_arc (
             int x,
             int y,
             int radius,
             int start,
             int end,
             Color color )
```

### 4.26.1.6 renderer_draw_bezier()

```
void renderer_draw_bezier (
             const short * vx,
             const short * vy,
             int n,
             int s,
             Color color )
```

### 4.26.1.7 renderer_draw_circle()

```
void renderer_draw_circle (
             int x,
             int y,
             int radius,
             Color color )
```

### 4.26.1.8 renderer_draw_ellipse()

```
void renderer_draw_ellipse (
             int x,
             int y,
             int rx,
             int ry,
             Color color )
```

### 4.26.1.9 renderer_draw_filled_circle()

```
void renderer_draw_filled_circle (
             int x,
             int y,
             int radius,
             Color color )
```

#### 4.26.1.10 renderer_draw_filled_ellipse()

```
void renderer_draw_filled_ellipse (
             int x,
             int y,
             int rx,
             int ry,
             Color color )
```

#### 4.26.1.11 renderer_draw_filled_pie()

```
void renderer_draw_filled_pie (
             int x,
             int y,
             int radius,
             int start,
             int end,
             Color color )
```

#### 4.26.1.12 renderer_draw_filled_polygon()

```
void renderer_draw_filled_polygon (
             const short * vx,
             const short * vy,
             int n,
             Color color )
```

#### 4.26.1.13 renderer_draw_filled_rect()

```
void renderer_draw_filled_rect (
             int x,
             int y,
             int width,
             int height,
             Color color )
```

#### 4.26.1.14 renderer_draw_filled_rounded_rect()

```
void renderer_draw_filled_rounded_rect (
             int x,
             int y,
             int width,
             int height,
             int radius,
             Color color )
```

#### 4.26.1.15 renderer_draw_filled_triangle()

```
void renderer_draw_filled_triangle (
             int x1,
             int y1,
             int x2,
             int y2,
             int x3,
             int y3,
             Color color )
```

#### 4.26.1.16 renderer_draw_line()

```
void renderer_draw_line (
             int x1,
             int y1,
             int x2,
             int y2,
             int thickness,
             Color color )
```

#### 4.26.1.17 renderer_draw_pie()

```
void renderer_draw_pie (
             int x,
             int y,
             int radius,
             int start,
             int end,
             Color color )
```

#### 4.26.1.18 renderer_draw_pixel()

```
void renderer_draw_pixel (
             int x,
             int y,
             Color color )
```

#### 4.26.1.19 renderer_draw_polygon()

```
void renderer_draw_polygon (
             const short * vx,
             const short * vy,
             int n,
             Color color )
```

#### 4.26.1.20 renderer_draw_rect()

```
void renderer_draw_rect (
            int x,
            int y,
            int width,
            int height,
            Color color )
```

#### 4.26.1.21 renderer_draw_rounded_rect()

```
void renderer_draw_rounded_rect (
            int x,
            int y,
            int width,
            int height,
            int radius,
            Color color )
```

#### 4.26.1.22 renderer_draw_text()

```
void renderer_draw_text (
            const char * text,
            int x,
            int y,
            Color color )
```

#### 4.26.1.23 renderer_draw_texture()

```
void renderer_draw_texture (
            Texture * texture,
            int x,
            int y,
            int width,
            int height )
```

#### 4.26.1.24 renderer_draw_triangle()

```
void renderer_draw_triangle (
            int x1,
            int y1,
            int x2,
            int y2,
            int x3,
            int y3,
            Color color )
```

#### 4.26.1.25 renderer_query_text_size()

```
SDL_Point renderer_query_text_size (
            const char * text )
```

#### 4.26.1.26 renderer_reset_clip_rect()

```
void renderer_reset_clip_rect ( )
```

#### 4.26.1.27 renderer_resize_framebuffer()

```
void renderer_resize_framebuffer (
            Texture * framebuffer,
            int width,
            int height )
```

#### 4.26.1.28 renderer_set_clip_rect()

```
void renderer_set_clip_rect (
            int x,
            int y,
            int width,
            int height )
```

#### 4.26.1.29 renderer_set_default_font()

```
void renderer_set_default_font (
            Font * font )
```

## 4.27 renderer.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
    #include <SDL2_gfxPrimitives.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
    #include <SDL2/SDL2_gfxPrimitives.h>
#endif

#include "../color/color.h"
#include "../font/font.h"
#include "../texture/texture.h"

void renderer_set_default_font(Font* font);
void renderer_set_clip_rect(int x, int y, int width, int height);
void renderer_reset_clip_rect();

Texture* renderer_create_framebuffer(int width, int height);
void renderer_resize_framebuffer(Texture* framebuffer, int width, int height);
void renderer_bind_framebuffer(Texture* framebuffer);

void renderer_clear(Color color);
void renderer_draw_pixel(int x, int y, Color color);
void renderer_draw_line(int x1, int y1, int x2, int y2, int thickness, Color color);
void renderer_draw_rect(int x, int y, int width, int height, Color color);
void renderer_draw_filled_rect(int x, int y, int width, int height, Color color);
void renderer_draw_circle(int x, int y, int radius, Color color);
void renderer_draw_filled_circle(int x, int y, int radius, Color color);
void renderer_draw_ellipse(int x, int y, int rx, int ry, Color color);
void renderer_draw_filled_ellipse(int x, int y, int rx, int ry, Color color);
void renderer_draw_triangle(int x1, int y1, int x2, int y2, int x3, int y3, Color color);
void renderer_draw_filled_triangle(int x1, int y1, int x2, int y2, int x3, int y3, Color color);
void renderer_draw_rounded_rect(int x, int y, int width, int height, int radius, Color color);
void renderer_draw_filled_rounded_rect(int x, int y, int width, int height, int radius, Color color);
void renderer_draw_polygon(const short* vx, const short* vy, int n, Color color);
void renderer_draw_filled_polygon(const short* vx, const short* vy, int n, Color color);
void renderer_draw_arc(int x, int y, int radius, int start, int end, Color color);
void renderer_draw_pie(int x, int y, int radius, int start, int end, Color color);
void renderer_draw_filled_pie(int x, int y, int radius, int start, int end, Color color);
void renderer_draw_bezier(const short* vx, const short* vy, int n, int s, Color color);
void renderer_draw_texture(Texture* texture, int x, int y, int width, int height);
void renderer_draw_text(const char* text, int x, int y, Color color);
SDL_Point renderer_query_text_size(const char* text);

void _renderer_set_target(SDL_Renderer* renderer);
```

## 4.28 src/texture/texture.c File Reference

```
#include "texture.h"
#include "../utils/vector/vector.h"
```

#### Functions

- Texture ∗ texture_load (SDL_Renderer ∗renderer, const char ∗path)
- void _texture_init ()
- void _texture_add (Texture ∗texture)
- void _texture_close ()

#### Variables

- Vector ∗ textures

### 4.28.1 Function Documentation

#### 4.28.1.1 _texture_add()

```
void _texture_add (
            Texture ∗ texture )
```

#### 4.28.1.2 _texture_close()

```
void _texture_close ( )
```

#### 4.28.1.3 _texture_init()

```
void _texture_init ( )
```

#### 4.28.1.4 texture_load()

```
Texture ∗ texture_load (
            SDL_Renderer ∗ renderer,
            const char ∗ path )
```

### 4.28.2 Variable Documentation

#### 4.28.2.1 textures

```
Vector* textures
```

## 4.29 src/texture/texture.h File Reference

### Classes

- struct Texture

### Typedefs

- typedef struct Texture Texture

### Functions

- Texture ∗ texture_load (SDL_Renderer ∗renderer, const char ∗path)
- void _texture_init ()
- void _texture_add (Texture ∗texture)
- void _texture_close ()

### 4.29.1 Typedef Documentation

#### 4.29.1.1 Texture

```
typedef struct Texture Texture
```

### 4.29.2 Function Documentation

#### 4.29.2.1 _texture_add()

```
void _texture_add (
            Texture * texture )
```

#### 4.29.2.2 _texture_close()

```
void _texture_close ( )
```

#### 4.29.2.3 _texture_init()

```
void _texture_init ( )
```

### 4.29.2.4 texture_load()

```
Texture * texture_load (
              SDL_Renderer * renderer,
              const char * path )
```

## 4.30 texture.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
    #include <SDL_image.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
    #include <SDL2/SDL_image.h>
#endif

typedef struct Texture
{
    SDL_Texture* texture;
    int width;
    int height;
} Texture;

Texture* texture_load(SDL_Renderer* renderer, const char* path);

//internal functions
void _texture_init();
void _texture_add(Texture* texture);
void _texture_close();
```

## 4.31 src/ui/ui.c File Reference

```
#include "ui.h"
#include "../app/app.h"
```

### Functions

- void _ui_init (UIData *ui_data, int width, int height)
- void _ui_handle_event (UIData *ui_data, SDL_Event *event)
- void _ui_update (UIData *ui_data)
- void _ui_render (UIData *ui_data)
- void _ui_close (UIData *ui_data)
- void _ui_set_target (UIData *ui_data)
- UIData * _ui_get_target ()

### Variables

- UIData * target_ui_data = NULL

## 4.31.1 Function Documentation

### 4.31.1.1 _ui_close()

```
void _ui_close (
             UIData * ui_data )
```

### 4.31.1.2 _ui_get_target()

```
UIData * _ui_get_target ( )
```

### 4.31.1.3 _ui_handle_event()

```
void _ui_handle_event (
             UIData * ui_data,
             SDL_Event * event )
```

### 4.31.1.4 _ui_init()

```
void _ui_init (
             UIData * ui_data,
             int width,
             int height )
```

### 4.31.1.5 _ui_render()

```
void _ui_render (
             UIData * ui_data )
```

### 4.31.1.6 _ui_set_target()

```
void _ui_set_target (
             UIData * ui_data )
```

### 4.31.1.7 _ui_update()

```
void _ui_update (
             UIData * ui_data )
```

## 4.31.2 Variable Documentation

### 4.31.2.1 target_ui_data

```
UIData* target_ui_data = NULL
```

## 4.32 src/ui/ui.h File Reference

```
#include "ui_element/ui_element.h"
```

### Classes

- struct UIData

### Typedefs

- typedef struct UIData UIData

### Functions

- void _ui_init (UIData ∗ui_data, int width, int height)
- void _ui_handle_event (UIData ∗ui_data, SDL_Event ∗event)
- void _ui_update (UIData ∗ui_data)
- void _ui_render (UIData ∗ui_data)
- void _ui_close (UIData ∗ui_data)
- void _ui_set_target (UIData ∗ui_data)
- UIData ∗ _ui_get_target ()

### 4.32.1 Typedef Documentation

#### 4.32.1.1 UIData

```
typedef struct UIData UIData
```

### 4.32.2 Function Documentation

#### 4.32.2.1 _ui_close()

```
void _ui_close (
            UIData ∗ ui_data )
```

#### 4.32.2.2 _ui_get_target()

```
UIData ∗ _ui_get_target ( )
```

#### 4.32.2.3 _ui_handle_event()

```
void _ui_handle_event (
            UIData ∗ ui_data,
            SDL_Event ∗ event )
```

#### 4.32.2.4 _ui_init()

```
void _ui_init (
            UIData * ui_data,
            int width,
            int height )
```

#### 4.32.2.5 _ui_render()

```
void _ui_render (
            UIData * ui_data )
```

#### 4.32.2.6 _ui_set_target()

```
void _ui_set_target (
            UIData * ui_data )
```

#### 4.32.2.7 _ui_update()

```
void _ui_update (
            UIData * ui_data )
```

## 4.33 ui.h

Go to the documentation of this file.

```
#pragma once

#include "ui_element/ui_element.h"

typedef struct UIData
{
    UIContainer* main_container;
    char text_input[SDL_TEXTINPUTEVENT_TEXT_SIZE];
    bool backspace_pressed;
    bool mouse_captured;
    UISplitButton* expanded_splitbutton;
} UIData;

//internal functions
void _ui_init(UIData* ui_data, int width, int height);
void _ui_handle_event(UIData* ui_data, SDL_Event* event);
void _ui_update(UIData* ui_data);
void _ui_render(UIData* ui_data);
void _ui_close(UIData* ui_data);
void _ui_set_target(UIData* ui_data);
UIData* _ui_get_target();
```

## 4.34 src/ui/ui_constraint/ui_constraint.c File Reference

```
#include "ui_constraint.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

### Functions

- UIConstraint new_pixel_constraint (int value)
- UIConstraint new_center_constraint ()
- UIConstraint new_relative_constraint (double value)
- UIConstraint new_offset_constraint (double value)
- UIConstraint new_aspect_constraint (double value)
- UIConstraints constraints_from_string (const char ∗string)

## 4.34.1 Function Documentation

### 4.34.1.1 constraints_from_string()

```
UIConstraints constraints_from_string (
             const char * string )
```

### 4.34.1.2 new_aspect_constraint()

```
UIConstraint new_aspect_constraint (
             double value )
```

### 4.34.1.3 new_center_constraint()

```
UIConstraint new_center_constraint ( )
```

### 4.34.1.4 new_offset_constraint()

```
UIConstraint new_offset_constraint (
             double value )
```

### 4.34.1.5 new_pixel_constraint()

```
UIConstraint new_pixel_constraint (
             int value )
```

### 4.34.1.6 new_relative_constraint()

```
UIConstraint new_relative_constraint (
             double value )
```

# 4.35 src/ui/ui_constraint/ui_constraint.h File Reference

### Classes

- struct UIConstraint
- struct UIConstraints

### Typedefs

- typedef enum ConstraintType ConstraintType
- typedef struct UIConstraint UIConstraint
- typedef struct UIConstraints UIConstraints

### Enumerations

- enum ConstraintType {
  CT_PIXEL = 0 , CT_CENTER , CT_RELATIVE , CT_OFFSET ,
  CT_ASPECT }

### Functions

- UIConstraint new_pixel_constraint (int value)
- UIConstraint new_center_constraint ()
- UIConstraint new_relative_constraint (double value)
- UIConstraint new_offset_constraint (double value)
- UIConstraint new_aspect_constraint (double value)
- UIConstraints constraints_from_string (const char ∗string)

## 4.35.1 Typedef Documentation

### 4.35.1.1 ConstraintType

```
typedef enum ConstraintType ConstraintType
```

### 4.35.1.2 UIConstraint

```
typedef struct UIConstraint UIConstraint
```

### 4.35.1.3 UIConstraints

```
typedef struct UIConstraints UIConstraints
```

## 4.35.2 Enumeration Type Documentation

### 4.35.2.1 ConstraintType

```
enum ConstraintType
```

**Enumerator**

| | |
|---|---|
| CT_PIXEL | |
| CT_CENTER | |
| CT_RELATIVE | |
| CT_OFFSET | |
| CT_ASPECT | |

## 4.35.3 Function Documentation

### 4.35.3.1 constraints_from_string()

```
UIConstraints constraints_from_string (
             const char * string )
```

### 4.35.3.2 new_aspect_constraint()

```
UIConstraint new_aspect_constraint (
             double value )
```

### 4.35.3.3 new_center_constraint()

```
UIConstraint new_center_constraint ( )
```

### 4.35.3.4 new_offset_constraint()

```
UIConstraint new_offset_constraint (
             double value )
```

### 4.35.3.5 new_pixel_constraint()

```
UIConstraint new_pixel_constraint (
             int value )
```

### 4.35.3.6 new_relative_constraint()

```
UIConstraint new_relative_constraint (
             double value )
```

# 4.36 ui_constraint.h

Go to the documentation of this file.

```
#pragma once

typedef enum ConstraintType
{
    CT_PIXEL = 0,
    CT_CENTER,
    CT_RELATIVE,
    CT_OFFSET,
    CT_ASPECT
} ConstraintType;

typedef struct UIConstraint
{
    double value;
    ConstraintType constraint_type;
    void (*recalculate)(void* self);
} UIConstraint;

typedef struct UIConstraints
{
    UIConstraint x, y, width, height;
} UIConstraints;

//API functions
UIConstraint new_pixel_constraint(int value);
UIConstraint new_center_constraint();
UIConstraint new_relative_constraint(double value);
UIConstraint new_offset_constraint(double value);
UIConstraint new_aspect_constraint(double value);
UIConstraints constraints_from_string(const char* string);
```

## 4.37 src/ui/ui_element/ui_element.c File Reference

```
#include "ui_element.h"
#include "../../app/app.h"
#include "../../renderer/renderer.h"
#include "../../input/input.h"
#include "../../utils/math/math.h"
#include <string.h>
```

### Classes

- struct _UIDropdownItem
- struct _UISplitButtonItem

### Typedefs

- typedef struct _UIDropdownItem _UIDropdownItem
- typedef struct _UISplitButtonItem _UISplitButtonItem

### Functions

- UIContainer * ui_create_container (UIContainer *parent, UIConstraints constraints, void(*on_size_changed)(UIContainer *self, SDL_Point size))
- UIPanel * ui_create_panel (UIContainer *parent, UIConstraints constraints, Color color, Color border_color, Uint32 border_width, Uint32 roundness)
- UILabel * ui_create_label (UIContainer *parent, UIConstraints constraints, const char *text, Color color)
- UIButton * ui_create_button (UIContainer *parent, UIConstraints constraints, const char *text, Color color, Color text_color, void(*on_click)(UIButton *self))
- UIImageButton * ui_create_imagebutton (UIContainer *parent, UIConstraints constraints, Texture *texture, void(*on_click)(UIImageButton *self))
- UITextbox * ui_create_textbox (UIContainer *parent, UIConstraints constraints, const char *text, Color color, Color text_color, void(*on_text_changed)(UITextbox *self, const char *text))
- UICheckbox * ui_create_checkbox (UIContainer *parent, UIConstraints constraints, Color checked_color, Color unchecked_color, void(*on_checked_changed)(UICheckbox *self, bool checked))
- UISlider * ui_create_slider (UIContainer *parent, UIConstraints constraints, double value, Color color, Color slider_color, void(*on_value_changed)(UISlider *self, double value))
- UIDropdownList * ui_create_dropdown (UIContainer *parent, UIConstraints constraints, char *items, Color color, Color text_color, void(*on_selection_changed)(UIDropdownList *self, Sint32 index))
- UISplitButton * ui_create_splitbutton (UIContainer *parent, UIConstraints constraints, char *items, Color color, Color text_color, void(*on_item_clicked)(UISplitButton *self, Sint32 index), bool auto_dropdown)
- void _ui_container_update (UIElement *self)
- void _ui_container_recalculate (UIElement *sibling, UIElement *self)
- void _ui_container_render (UIElement *self)
- void _ui_container_destroy (UIElement *self)

### 4.37.1 Typedef Documentation

#### 4.37.1.1 _UIDropdownItem

```
typedef struct _UIDropdownItem _UIDropdownItem
```

#### 4.37.1.2 _UISplitButtonItem

```
typedef struct _UISplitButtonItem _UISplitButtonItem
```

### 4.37.2 Function Documentation

#### 4.37.2.1 _ui_container_destroy()

```
void _ui_container_destroy (
            UIElement * self )
```

#### 4.37.2.2 _ui_container_recalculate()

```
void _ui_container_recalculate (
            UIElement * sibling,
            UIElement * self )
```

#### 4.37.2.3 _ui_container_render()

```
void _ui_container_render (
            UIElement * self )
```

#### 4.37.2.4 _ui_container_update()

```
void _ui_container_update (
            UIElement * self )
```

#### 4.37.2.5 ui_create_button()

```
UIButton * ui_create_button (
            UIContainer * parent,
            UIConstraints constraints,
            const char * text,
            Color color,
            Color text_color,
            void(*)(UIButton *self) on_click )
```

#### 4.37.2.6 ui_create_checkbox()

```
UICheckbox * ui_create_checkbox (
            UIContainer * parent,
            UIConstraints constraints,
            Color checked_color,
            Color unchecked_color,
            void(*)(UICheckbox *self, bool checked) on_checked_changed )
```

#### 4.37.2.7 ui_create_container()

```
UIContainer * ui_create_container (
             UIContainer * parent,
             UIConstraints constraints,
             void(*)(UIContainer *self, SDL_Point size) on_size_changed )
```

#### 4.37.2.8 ui_create_dropdown()

```
UIDropdownList * ui_create_dropdown (
             UIContainer * parent,
             UIConstraints constraints,
             char * items,
             Color color,
             Color text_color,
             void(*)(UIDropdownList *self, Sint32 index) on_selection_changed )
```

#### 4.37.2.9 ui_create_imagebutton()

```
UIImageButton * ui_create_imagebutton (
             UIContainer * parent,
             UIConstraints constraints,
             Texture * texture,
             void(*)(UIImageButton *self) on_click )
```

#### 4.37.2.10 ui_create_label()

```
UILabel * ui_create_label (
             UIContainer * parent,
             UIConstraints constraints,
             const char * text,
             Color color )
```

#### 4.37.2.11 ui_create_panel()

```
UIPanel * ui_create_panel (
             UIContainer * parent,
             UIConstraints constraints,
             Color color,
             Color border_color,
             Uint32 border_width,
             Uint32 roundness )
```

#### 4.37.2.12 ui_create_slider()

```
UISlider * ui_create_slider (
             UIContainer * parent,
             UIConstraints constraints,
             double value,
             Color color,
             Color slider_color,
             void(*)(UISlider *self, double value) on_value_changed )
```

#### 4.37.2.13 ui_create_splitbutton()

```
UISplitButton * ui_create_splitbutton (
            UIContainer * parent,
            UIConstraints constraints,
            char * items,
            Color color,
            Color text_color,
            void(*)(UISplitButton *self, Sint32 index) on_item_clicked,
            bool auto_dropdown )
```

#### 4.37.2.14 ui_create_textbox()

```
UITextbox * ui_create_textbox (
            UIContainer * parent,
            UIConstraints constraints,
            const char * text,
            Color color,
            Color text_color,
            void(*)(UITextbox *self, const char *text) on_text_changed )
```

## 4.38 src/ui/ui_element/ui_element.h File Reference

```
#include "../ui_constraint/ui_constraint.h"
#include "../../utils/vector/vector.h"
#include "../../color/color.h"
#include "../../texture/texture.h"
#include <stdbool.h>
```

### Classes

- struct UIElement
- struct UIContainer
- struct UIPanel
- struct UILabel
- struct UIButton
- struct UIImageButton
- struct UITextbox
- struct UICheckbox
- struct UISlider
- struct UIDropdownList
- struct UISplitButton

### Macros

- #define UITEXT_MAX_LENGTH 50

### Typedefs

- typedef struct UIElement UIElement
- typedef struct UIContainer UIContainer
- typedef struct UIPanel UIPanel
- typedef struct UILabel UILabel
- typedef enum MouseState MouseState
- typedef struct UIButton UIButton
- typedef struct UIImageButton UIImageButton
- typedef struct UITextbox UITextbox
- typedef struct UICheckbox UICheckbox
- typedef struct UISlider UISlider
- typedef struct UIDropdownList UIDropdownList
- typedef struct UISplitButton UISplitButton

### Enumerations

- enum MouseState { MS_NONE = 0 , MS_HOVER , MS_PRESS }

### Functions

- UIContainer ∗ ui_create_container (UIContainer ∗parent, UIConstraints constraints, void(∗on_size_↩ changed)(UIContainer ∗self, SDL_Point size))
- UIPanel ∗ ui_create_panel (UIContainer ∗parent, UIConstraints constraints, Color color, Color border_color, Uint32 border_width, Uint32 roundness)
- UILabel ∗ ui_create_label (UIContainer ∗parent, UIConstraints constraints, const char ∗text, Color color)
- UIButton ∗ ui_create_button (UIContainer ∗parent, UIConstraints constraints, const char ∗text, Color color, Color text_color, void(∗on_click)(UIButton ∗self))
- UIImageButton ∗ ui_create_imagebutton (UIContainer ∗parent, UIConstraints constraints, Texture ∗texture, void(∗on_click)(UIImageButton ∗self))
- UITextbox ∗ ui_create_textbox (UIContainer ∗parent, UIConstraints constraints, const char ∗text, Color color, Color text_color, void(∗on_text_changed)(UITextbox ∗self, const char ∗text))
- UICheckbox ∗ ui_create_checkbox (UIContainer ∗parent, UIConstraints constraints, Color checked_color, Color unchecked_color, void(∗on_checked_changed)(UICheckbox ∗self, bool checked))
- UISlider ∗ ui_create_slider (UIContainer ∗parent, UIConstraints constraints, double value, Color color, Color slider_color, void(∗on_value_changed)(UISlider ∗self, double value))
- UIDropdownList ∗ ui_create_dropdown (UIContainer ∗parent, UIConstraints constraints, char ∗items, Color color, Color text_color, void(∗on_selection_changed)(UIDropdownList ∗self, Sint32 index))
- UISplitButton ∗ ui_create_splitbutton (UIContainer ∗parent, UIConstraints constraints, char ∗items, Color color, Color text_color, void(∗on_item_clicked)(UISplitButton ∗self, Sint32 index), bool auto_dropdown)
- void _ui_container_update (UIElement ∗self)
- void _ui_container_recalculate (UIElement ∗sibling, UIElement ∗self)
- void _ui_container_render (UIElement ∗self)
- void _ui_container_destroy (UIElement ∗self)

## 4.38.1 Macro Definition Documentation

### 4.38.1.1 UITEXT_MAX_LENGTH

```
#define UITEXT_MAX_LENGTH 50
```

## 4.38.2 Typedef Documentation

### 4.38.2.1 MouseState

```
typedef enum MouseState MouseState
```

### 4.38.2.2 UIButton

```
typedef struct UIButton UIButton
```

### 4.38.2.3 UICheckbox

```
typedef struct UICheckbox UICheckbox
```

### 4.38.2.4 UIContainer

```
typedef struct UIContainer UIContainer
```

### 4.38.2.5 UIDropdownList

```
typedef struct UIDropdownList UIDropdownList
```

### 4.38.2.6 UIElement

```
typedef struct UIElement UIElement
```

### 4.38.2.7 UIImageButton

```
typedef struct UIImageButton UIImageButton
```

### 4.38.2.8 UILabel

```
typedef struct UILabel UILabel
```

### 4.38.2.9 UIPanel

```
typedef struct UIPanel UIPanel
```

### 4.38.2.10 UISlider

```
typedef struct UISlider UISlider
```

#### 4.38.2.11 UISplitButton

```
typedef struct UISplitButton UISplitButton
```

#### 4.38.2.12 UITextbox

```
typedef struct UITextbox UITextbox
```

### 4.38.3 Enumeration Type Documentation

#### 4.38.3.1 MouseState

```
enum MouseState
```

**Enumerator**

| | |
|---|---|
| MS_NONE | |
| MS_HOVER | |
| MS_PRESS | |

### 4.38.4 Function Documentation

#### 4.38.4.1 _ui_container_destroy()

```
void _ui_container_destroy (
            UIElement * self )
```

#### 4.38.4.2 _ui_container_recalculate()

```
void _ui_container_recalculate (
            UIElement * sibling,
            UIElement * self )
```

#### 4.38.4.3 _ui_container_render()

```
void _ui_container_render (
            UIElement * self )
```

#### 4.38.4.4 _ui_container_update()

```
void _ui_container_update (
            UIElement * self )
```

#### 4.38.4.5 ui_create_button()

```
UIButton * ui_create_button (
             UIContainer * parent,
             UIConstraints constraints,
             const char * text,
             Color color,
             Color text_color,
             void(*)(UIButton *self) on_click )
```

#### 4.38.4.6 ui_create_checkbox()

```
UICheckbox * ui_create_checkbox (
             UIContainer * parent,
             UIConstraints constraints,
             Color checked_color,
             Color unchecked_color,
             void(*)(UICheckbox *self, bool checked) on_checked_changed )
```

#### 4.38.4.7 ui_create_container()

```
UIContainer * ui_create_container (
             UIContainer * parent,
             UIConstraints constraints,
             void(*)(UIContainer *self, SDL_Point size) on_size_changed )
```

#### 4.38.4.8 ui_create_dropdown()

```
UIDropdownList * ui_create_dropdown (
             UIContainer * parent,
             UIConstraints constraints,
             char * items,
             Color color,
             Color text_color,
             void(*)(UIDropdownList *self, Sint32 index) on_selection_changed )
```

#### 4.38.4.9 ui_create_imagebutton()

```
UIImageButton * ui_create_imagebutton (
             UIContainer * parent,
             UIConstraints constraints,
             Texture * texture,
             void(*)(UIImageButton *self) on_click )
```

#### 4.38.4.10 ui_create_label()

```
UILabel * ui_create_label (
             UIContainer * parent,
             UIConstraints constraints,
             const char * text,
             Color color )
```

#### 4.38.4.11 ui_create_panel()

```
UIPanel * ui_create_panel (
              UIContainer * parent,
              UIConstraints constraints,
              Color color,
              Color border_color,
              Uint32 border_width,
              Uint32 roundness )
```

#### 4.38.4.12 ui_create_slider()

```
UISlider * ui_create_slider (
              UIContainer * parent,
              UIConstraints constraints,
              double value,
              Color color,
              Color slider_color,
              void(*)(UISlider *self, double value) on_value_changed )
```

#### 4.38.4.13 ui_create_splitbutton()

```
UISplitButton * ui_create_splitbutton (
              UIContainer * parent,
              UIConstraints constraints,
              char * items,
              Color color,
              Color text_color,
              void(*)(UISplitButton *self, Sint32 index) on_item_clicked,
              bool auto_dropdown )
```

#### 4.38.4.14 ui_create_textbox()

```
UITextbox * ui_create_textbox (
              UIContainer * parent,
              UIConstraints constraints,
              const char * text,
              Color color,
              Color text_color,
              void(*)(UITextbox *self, const char *text) on_text_changed )
```

## 4.39 ui_element.h

Go to the documentation of this file.

```
#pragma once

#ifdef _WIN32
    #include <SDL.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
#endif

#include "../ui_constraint/ui_constraint.h"
```

```
#include "../../utils/vector/vector.h"
#include "../../color/color.h"
#include "../../texture/texture.h"

#include <stdbool.h>

#define UITEXT_MAX_LENGTH 50

typedef struct UIElement UIElement;
typedef struct UIElement
{
    UIElement* parent;
    UIConstraints constraints;
    SDL_Point position;
    SDL_Point size;

    void (*update)(UIElement* self);
    void (*recalculate)(UIElement* sibling, UIElement* self);
    void (*render)(UIElement* self);
    void (*destroy)(UIElement* self);
} UIElement;

typedef struct UIContainer UIContainer;
typedef struct UIContainer
{
    UIElement base;

    Vector* children;
    void (*on_size_changed)(UIContainer* self, SDL_Point size);
} UIContainer;

typedef struct UIPanel
{
    UIElement base;

    Color color;
    Color border_color;
    Uint32 border_width;
    Uint32 corner_radius;
} UIPanel;

typedef struct UILabel
{
    UIElement base;

    char text[UITEXT_MAX_LENGTH + 1];
    Color color;
} UILabel;

typedef enum MouseState { MS_NONE = 0, MS_HOVER, MS_PRESS } MouseState;
typedef struct UIButton UIButton;
typedef struct UIButton
{
    UIElement base;

    char text[UITEXT_MAX_LENGTH + 1];
    SDL_Point text_position;
    Color color;
    Color text_color;
    Uint32 corner_radius;
    MouseState mouse_state;
    void (*on_click)(UIButton* self);
} UIButton;

typedef struct UIImageButton UIImageButton;
typedef struct UIImageButton
{
    UIElement base;

    Texture* texture;
    MouseState mouse_state;
    void (*on_click)(UIImageButton* self);
} UIImageButton;

typedef struct UITextbox UITextbox;
typedef struct UITextbox
{
    UIElement base;

    char text[UITEXT_MAX_LENGTH + 1];
    Color color;
    Color text_color;
    Uint32 corner_radius;
    bool focused;
    MouseState mouse_state;
    void (*on_text_changed)(UITextbox* self, const char* text);
} UITextbox;
```

```

typedef struct UICheckbox UICheckbox;
typedef struct UICheckbox
{
    UIElement base;

    bool checked;
    Color checked_color;
    Color unchecked_color;
    Uint32 corner_radius;
    MouseState mouse_state;
    void (*on_checked_changed)(UICheckbox* self, bool checked);
} UICheckbox;

typedef struct UISlider UISlider;
typedef struct UISlider
{
    UIElement base;

    double value;
    Color color;
    Color slider_color;
    Uint32 thickness;
    Uint32 corner_radius;
    MouseState mouse_state;
    void (*on_value_changed)(UISlider* self, double value);
} UISlider;

typedef struct UIDropdownList UIDropdownList;
typedef struct UIDropdownList
{
    UIElement base;

    Vector* items;
    Uint32 selected_item;
    bool expanded;
    Color color;
    Color text_color;
    Uint32 corner_radius;
    void (*on_selection_changed)(UIDropdownList* self, Sint32 index);
} UIDropdownList;

typedef struct UISplitButton UISplitButton;
typedef struct UISplitButton
{
    UIElement base;

    Vector* items;
    bool expanded;
    Color color;
    Color text_color;
    Uint32 corner_radius;
    void (*on_item_clicked)(UISplitButton* self, Sint32 index);
    bool auto_dropdown;
} UISplitButton;

//API functions
UIContainer* ui_create_container(UIContainer* parent, UIConstraints constraints, void (*on_size_changed)(UIContainer* self, SDL_Point size));
UIPanel* ui_create_panel(UIContainer* parent, UIConstraints constraints, Color color, Color border_color, Uint32 border_width, Uint32 roundness);
UILabel* ui_create_label(UIContainer* parent, UIConstraints constraints, const char* text, Color color);
UIButton* ui_create_button(UIContainer* parent, UIConstraints constraints, const char* text, Color color, Color text_color, void (*on_click)(UIButton* self));
UIImageButton* ui_create_imagebutton(UIContainer* parent, UIConstraints constraints, Texture* texture, void (*on_click)(UIImageButton* self));
UITextbox* ui_create_textbox(UIContainer* parent, UIConstraints constraints, const char* text, Color color, Color text_color, void (*on_text_changed)(UITextbox* self, const char* text));
UICheckbox* ui_create_checkbox(UIContainer* parent, UIConstraints constraints, Color checked_color, Color unchecked_color, void (*on_checked_changed)(UICheckbox* self, bool checked));
UISlider* ui_create_slider(UIContainer* parent, UIConstraints constraints, double value, Color color, Color slider_color, void (*on_value_changed)(UISlider* self, double value));
UIDropdownList* ui_create_dropdown(UIContainer* parent, UIConstraints constraints, char* items, Color color, Color text_color, void (*on_selection_changed)(UIDropdownList* self, Sint32 index));
UISplitButton* ui_create_splitbutton(UIContainer* parent, UIConstraints constraints, char* items, Color color, Color text_color, void (*on_item_clicked)(UISplitButton* self, Sint32 index), bool auto_dropdown);

//internal functions
void _ui_container_update(UIElement* self);
void _ui_container_recalculate(UIElement* sibling, UIElement* self);
void _ui_container_render(UIElement* self);
void _ui_container_destroy(UIElement* self);
```

## 4.40 src/utils/math/math.c File Reference

```
#include "math.h"
```

### Functions

- double deg_to_rad (double deg)
- double rad_to_deg (double rad)
- double clamp (double value, double min, double max)
- double lerp (double a, double b, double t)
- double map (double x, double min1, double max1, double min2, double max2)
- bool check_collision_point_rect (int px, int py, int rx, int ry, int rw, int rh)

### 4.40.1 Function Documentation

#### 4.40.1.1 check_collision_point_rect()

```
bool check_collision_point_rect (
             int px,
             int py,
             int rx,
             int ry,
             int rw,
             int rh )
```

#### 4.40.1.2 clamp()

```
double clamp (
             double value,
             double min,
             double max )
```

#### 4.40.1.3 deg_to_rad()

```
double deg_to_rad (
             double deg )
```

#### 4.40.1.4 lerp()

```
double lerp (
             double a,
             double b,
             double t )
```

#### 4.40.1.5 map()

```
double map (
             double x,
             double min1,
             double max1,
             double min2,
             double max2 )
```

#### 4.40.1.6 rad_to_deg()

```
double rad_to_deg (
             double rad )
```

## 4.41 src/utils/math/math.h File Reference

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

**Macros**

- #define PI 3.14159265358979323846
- #define TWO_PI 6.28318530717958647692
- #define HALF_PI 1.57079632679489661923

**Functions**

- double deg_to_rad (double deg)
- double rad_to_deg (double rad)
- double clamp (double x, double min, double max)
- double lerp (double a, double b, double t)
- double map (double x, double min1, double max1, double min2, double max2)
- bool check_collision_point_rect (int px, int py, int rx, int ry, int rw, int rh)

### 4.41.1 Macro Definition Documentation

#### 4.41.1.1 HALF_PI

```
#define HALF_PI 1.57079632679489661923
```

#### 4.41.1.2 PI

```
#define PI 3.14159265358979323846
```

#### 4.41.1.3 TWO_PI

```
#define TWO_PI 6.28318530717958647692
```

## 4.41.2 Function Documentation

#### 4.41.2.1 check_collision_point_rect()

```
bool check_collision_point_rect (
             int px,
             int py,
             int rx,
             int ry,
             int rw,
             int rh )
```

#### 4.41.2.2 clamp()

```
double clamp (
             double x,
             double min,
             double max )
```

#### 4.41.2.3 deg_to_rad()

```
double deg_to_rad (
             double deg )
```

#### 4.41.2.4 lerp()

```
double lerp (
             double a,
             double b,
             double t )
```

#### 4.41.2.5 map()

```
double map (
             double x,
             double min1,
             double max1,
             double min2,
             double max2 )
```

#### 4.41.2.6 rad_to_deg()

```
double rad_to_deg (
             double rad )
```

## 4.42 math.h

Go to the documentation of this file.

```
#pragma once

#include <stdbool.h>

#define PI 3.14159265358979323846
#define TWO_PI 6.28318530717958647692
#define HALF_PI 1.57079632679489661923

double deg_to_rad(double deg);
double rad_to_deg(double rad);
double clamp(double x, double min, double max);
double lerp(double a, double b, double t);
double map(double x, double min1, double max1, double min2, double max2);

bool check_collision_point_rect(int px, int py, int rx, int ry, int rw, int rh);
```

## 4.43 src/utils/vector/vector.c File Reference

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

### Functions

- Vector ∗ vector_create (size_t capacity)
- void vector_free (Vector ∗vector)
- void ∗ vector_get (Vector ∗vector, size_t idx)
- void vector_push_back (Vector ∗vector, void ∗value)
- void vector_pop_back (Vector ∗vector)
- void vector_insert (Vector ∗vector, size_t idx, void ∗value)
- bool vector_contains (Vector ∗vector, void ∗value)
- void vector_remove_at (Vector ∗vector, size_t idx)
- void vector_remove (Vector ∗vector, void ∗value)
- void vector_reserve (Vector ∗vector, size_t capacity)
- size_t vector_size (Vector ∗vector)
- void vector_clear (Vector ∗vector)

### 4.43.1 Function Documentation

#### 4.43.1.1 vector_clear()

```
void vector_clear (
            Vector ∗ vector )
```

#### 4.43.1.2 vector_contains()

```
bool vector_contains (
            Vector ∗ vector,
            void ∗ value )
```

#### 4.43.1.3 vector_create()

```
Vector * vector_create (
             size_t capacity )
```

#### 4.43.1.4 vector_free()

```
void vector_free (
             Vector * vector )
```

#### 4.43.1.5 vector_get()

```
void * vector_get (
             Vector * vector,
             size_t idx )
```

#### 4.43.1.6 vector_insert()

```
void vector_insert (
             Vector * vector,
             size_t idx,
             void * value )
```

#### 4.43.1.7 vector_pop_back()

```
void vector_pop_back (
             Vector * vector )
```

#### 4.43.1.8 vector_push_back()

```
void vector_push_back (
             Vector * vector,
             void * value )
```

#### 4.43.1.9 vector_remove()

```
void vector_remove (
             Vector * vector,
             void * value )
```

#### 4.43.1.10 vector_remove_at()

```
void vector_remove_at (
             Vector * vector,
             size_t idx )
```

#### 4.43.1.11 vector_reserve()

```
void vector_reserve (
            Vector * vector,
            size_t capacity )
```

#### 4.43.1.12 vector_size()

```
size_t vector_size (
            Vector * vector )
```

## 4.44 src/utils/vector/vector.h File Reference

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

### Classes

- struct Vector

### Typedefs

- typedef struct Vector Vector

### Functions

- Vector ∗ vector_create (size_t capacity)
- void vector_free (Vector ∗vector)
- void ∗ vector_get (Vector ∗vector, size_t idx)
- void vector_push_back (Vector ∗vector, void ∗value)
- void vector_pop_back (Vector ∗vector)
- void vector_insert (Vector ∗vector, size_t idx, void ∗value)
- bool vector_contains (Vector ∗vector, void ∗value)
- void vector_remove_at (Vector ∗vector, size_t idx)
- void vector_remove (Vector ∗vector, void ∗value)
- void vector_reserve (Vector ∗vector, size_t capacity)
- size_t vector_size (Vector ∗vector)
- void vector_clear (Vector ∗vector)

### 4.44.1 Typedef Documentation

#### 4.44.1.1 Vector

```
typedef struct Vector Vector
```

## 4.44.2 Function Documentation

### 4.44.2.1 vector_clear()

```
void vector_clear (
             Vector * vector )
```

### 4.44.2.2 vector_contains()

```
bool vector_contains (
             Vector * vector,
             void * value )
```

### 4.44.2.3 vector_create()

```
Vector * vector_create (
             size_t capacity )
```

### 4.44.2.4 vector_free()

```
void vector_free (
             Vector * vector )
```

### 4.44.2.5 vector_get()

```
void * vector_get (
             Vector * vector,
             size_t idx )
```

### 4.44.2.6 vector_insert()

```
void vector_insert (
             Vector * vector,
             size_t idx,
             void * value )
```

### 4.44.2.7 vector_pop_back()

```
void vector_pop_back (
             Vector * vector )
```

### 4.44.2.8 vector_push_back()

```
void vector_push_back (
             Vector * vector,
             void * value )
```

#### 4.44.2.9 vector_remove()

```
void vector_remove (
             Vector * vector,
             void * value )
```

#### 4.44.2.10 vector_remove_at()

```
void vector_remove_at (
             Vector * vector,
             size_t idx )
```

#### 4.44.2.11 vector_reserve()

```
void vector_reserve (
             Vector * vector,
             size_t capacity )
```

#### 4.44.2.12 vector_size()

```
size_t vector_size (
             Vector * vector )
```

## 4.45 vector.h

Go to the documentation of this file.

```
#pragma once

#include <stdlib.h>
#include <stdbool.h>

typedef struct Vector {
    size_t capacity;
    size_t size;
    void** data;
} Vector;

Vector* vector_create(size_t capacity);
void vector_free(Vector* vector);
void* vector_get(Vector* vector, size_t idx);
void vector_push_back(Vector* vector, void* value);
void vector_pop_back(Vector* vector);
void vector_insert(Vector* vector, size_t idx, void* value);
bool vector_contains(Vector* vector, void* value);
void vector_remove_at(Vector* vector, size_t idx);
void vector_remove(Vector* vector, void* value);
void vector_reserve(Vector* vector, size_t capacity);
size_t vector_size(Vector* vector);
void vector_clear(Vector* vector);
```

## 4.46 src/window/window.c File Reference

```
#include <assert.h>
#include "window.h"
#include "../app/app.h"
#include "../input/input.h"
#include "../renderer/renderer.h"
```

**Functions**

- Window ∗ window_create (const char ∗title, int width, int height, int flags)
- void window_show (Window ∗window)
- void window_hide (Window ∗window)
- void window_focus (Window ∗window)
- UIContainer ∗ window_get_main_container (Window ∗window)
- void _window_reset (Window ∗window)
- void _window_handle_event (Window ∗window, SDL_Event ∗event)
- void _window_update (Window ∗window)
- void _window_render (Window ∗window)
- void _window_close (Window ∗window)

## 4.46.1 Function Documentation

### 4.46.1.1 _window_close()

```
void _window_close (
            Window ∗ window )
```

### 4.46.1.2 _window_handle_event()

```
void _window_handle_event (
            Window ∗ window,
            SDL_Event ∗ event )
```

### 4.46.1.3 _window_render()

```
void _window_render (
            Window ∗ window )
```

### 4.46.1.4 _window_reset()

```
void _window_reset (
            Window ∗ window )
```

### 4.46.1.5 _window_update()

```
void _window_update (
            Window ∗ window )
```

### 4.46.1.6 window_create()

```
Window ∗ window_create (
            const char ∗ title,
            int width,
            int height,
            int flags )
```

#### 4.46.1.7 window_focus()

```
void window_focus (
             Window * window )
```

#### 4.46.1.8 window_get_main_container()

```
UIContainer * window_get_main_container (
             Window * window )
```

#### 4.46.1.9 window_hide()

```
void window_hide (
             Window * window )
```

#### 4.46.1.10 window_show()

```
void window_show (
             Window * window )
```

## 4.47 src/window/window.h File Reference

```
#include <stdbool.h>
#include "../input/input.h"
#include "../ui/ui.h"
#include "../ui/ui_element/ui_element.h"
```

### Classes

- struct Window

### Typedefs

- typedef struct Window Window

### Functions

- Window * window_create (const char *title, int width, int height, int flags)
- void window_show (Window *window)
- void window_hide (Window *window)
- void window_focus (Window *window)
- UIContainer * window_get_main_container (Window *window)
- void _window_reset (Window *window)
- void _window_handle_event (Window *window, SDL_Event *event)
- void _window_update (Window *window)
- void _window_render (Window *window)
- void _window_close (Window *window)

## 4.47.1 Typedef Documentation

### 4.47.1.1 Window

```
typedef struct Window Window
```

## 4.47.2 Function Documentation

### 4.47.2.1 _window_close()

```
void _window_close (
            Window * window )
```

### 4.47.2.2 _window_handle_event()

```
void _window_handle_event (
            Window * window,
            SDL_Event * event )
```

### 4.47.2.3 _window_render()

```
void _window_render (
            Window * window )
```

### 4.47.2.4 _window_reset()

```
void _window_reset (
            Window * window )
```

### 4.47.2.5 _window_update()

```
void _window_update (
            Window * window )
```

### 4.47.2.6 window_create()

```
Window * window_create (
            const char * title,
            int width,
            int height,
            int flags )
```

### 4.47.2.7 window_focus()

```
void window_focus (
            Window * window )
```

#### 4.47.2.8 window_get_main_container()

```
UIContainer * window_get_main_container (
              Window * window )
```

#### 4.47.2.9 window_hide()

```
void window_hide (
              Window * window )
```

#### 4.47.2.10 window_show()

```
void window_show (
              Window * window )
```

## 4.48 window.h

Go to the documentation of this file.

```
#ifndef WINDOW_H
#define WINDOW_H

#ifdef _WIN32
    #include <SDL.h>
#elif defined(__unix__) || defined(__linux__)
    #include <SDL2/SDL.h>
#endif

#include <stdbool.h>
#include "../input/input.h"
#include "../ui/ui.h"
#include "../ui/ui_element/ui_element.h"

typedef struct Window
{
    SDL_Window* window;
    SDL_Renderer* renderer;
    InputData input_data;
    UIData ui_data;
    bool close_requested;
} Window;

Window* window_create(const char* title, int width, int height, int flags);
void window_show(Window* window);
void window_hide(Window* window);
void window_focus(Window* window);
UIContainer* window_get_main_container(Window* window);

//API functions
void _window_reset(Window* window);
void _window_handle_event(Window* window, SDL_Event* event);
void _window_update(Window* window);
void _window_render(Window* window);
void _window_close(Window* window);

#endif
```

# Index