Game Engine

Generated by Doxygen 1.8.17

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 light Namespace Reference

Namespaces

- behavior

Classes

- class DirectionLight

  Light with a direction.
- class Light

  Base Light (Abstract)
- struct LightInfo

  Light Information.
- struct LightShader

  Light in the shader based on data retrieved from Light objects
- class PositionLight

  Light with a position and whose intensity can be attenuated.
- class SpotLight

  Light with a direction, a position and angles of diffusion.

5.2 light::behavior Namespace Reference

Classes

- class DirectionLightBehavior

  Behavior of a directed Light.
- class LightBehavior

  Behavior of a Light (Abstract)
- class PositionLightBehavior

  Behavior of a positionned Light.
- class SpotLightBehavior

  Behavior of a SpotLight.

5.3 material Namespace Reference

Classes

- class Material

  Base Material (Abstract)
- class MaterialColor

  Material with colors for the ambient, diffuse and specular components.
- class MaterialTexture

  Material with texture for the diffuse and specular components.

5.4 mesh Namespace Reference

Classes

- class LODMesh

  Mesh with Level of Details.
- class Mesh

  Base Mesh.
- struct MeshData

  Datas of a Mesh.

5.5 physics Namespace Reference

Namespaces

- bounding_box
- force
- ode
- rigid_body_behavior

Classes

- struct Collision

  Represents a collision between 2 BoundingBox.
- class PhysicsSystem

  Manage a whole physic system with RigidBody.
- class RigidBodyVolume

  Represents a rigid body with a volume (BoundingBox)

5.6 physics::bounding_box Namespace Reference

Classes

- class AABB

  BoundingBox aligned on axis.
- class BBFactory

  Factory of BoundingBox.
- class BoundingBox

  Represent a bounding box (Abstract)
- class OBB

  BoundingBox oriented.
- class RCBB

  Rectangle Cuboid BoundingBox (Abstract)
- class SphereBB

  Sphere BoundingBox.

Enumerations

- enum BB_TYPE { AABB_TYPE, OBB_TYPE, SPHEREBB_TYPE }

  Enum of the different types of BoundingBox.

5.6.1 Enumeration Type Documentation

5.6.1.1 BB_TYPE

```
enum physics::bounding_box::BB_TYPE
```

Enum of the different types of BoundingBox.

Enumerator

AABB_TYPE	
OBB_TYPE	
SPHEREBB_TYPE	

5.7 physics::force Namespace Reference

Classes

- class Force

  Represent a Force to apply on a RigidBodyVolume (Abstract)
- class GravityForce

  Force of the gravity.

5.8 physics::ode Namespace Reference

Classes

- class EulerODE

  Euler Ordinary Differential Equation.
- class ODE

  Ordinary Differential Equation (Abstract)
- class ODEFactory

  Factory of ODE.
- class RungeKutta4ODE

  Runge Kutta Ordinary Differential Equation.
- class VerletODE

  Verlet Ordinary Differential Equation.

Enumerations

- enum ODE_TYPE { EULER_TYPE, VERLET_TYPE, RK4_TYPE }

  Enum of the different types of ODE.

5.8.1 Enumeration Type Documentation

5.8.1.1 ODE_TYPE

```
enum physics::ode::ODE_TYPE
```

Enum of the different types of ODE.

Enumerator

EULER_TYPE	
VERLET_TYPE	
RK4_TYPE	

5.9 physics::rigid_body_behavior Namespace Reference

Classes

- class MoveDoorBehavior
- class MovementBehavior
- class RigidBodyBehavior
- class SwitchColorBehavior

Enumerations

- enum RigidBodyBehavior_TYPE { MovementBehavior_TYPE, SwitchColor_TYPE, MoveDoor_TYPE }

5.9.1 Enumeration Type Documentation

5.9.1.1 RigidBodyBehavior_TYPE

```
enum physics::rigid_body_behavior::RigidBodyBehavior_TYPE
```

Enumerator

MovementBehavior_TYPE	
SwitchColor_TYPE	
MoveDoor_TYPE	

5.10 scene Namespace Reference

Classes

- class BounceAABBScene

  Scene with Collisions.
- class BounceOBBScene

  Scene with Collisions.
- class BounceSphereBBScene

  Scene with Collisions.
- class LabScene

  Scene with Collisions.
- class Scene

  Base Scene (Abstract)
- class SceneLand

  Scene with a land and a ball on it.
- class ShadowedScene

  Scene with Collisions.
- class SolarSystem

  Scene with the sun, the earth, the moon end the sky.

5.11 scene_graph Namespace Reference

Classes

- class ElementSG

Base Element of the scene graph (Abstract)

- class NodeGameSG

  NodeSG that can have Meshes, light and camera.
- class NodeOnTopSG

  NodeGameSG on top of another.
- class NodeSG

  Base Node of the scene graph (Abstract)
- class RootSG

  Root of the scene graph.

5.12 shader Namespace Reference

Classes

- struct glsl_bool

  Vec3 used in shader.
- struct glsl_int

  Vec3 used in shader.
- struct glsl_mat4

  Mat4 used in shader.
- struct glsl_vec3

  Vec3 used in shader.
- class MainShaders
- class Shaders

  Group the VAOManager, the ShadersDataManager and the TextureManager.
- class ShadersDataManager

  Manager of the locations in shaders.
- class ShadowMap

  Represents a Shadow Map.
- class ShadowMapShaders

  Manager of the Shadow Mapping Shaders.
- class TextureManager

  Manager of all the texture in the shaders.
- class VAODataManager

  Manager of VBOs and EBO.

5.13 utils Namespace Reference

Classes

- struct Interval

  Represent an interval between 2 float values.
- struct Line

  Line with a start point and an end point.
- struct Plane

  Plane with a normal and a distance from the origin along the axis of the normal.
- struct Ray

  Ray with an origine and a direction.

Chapter 6

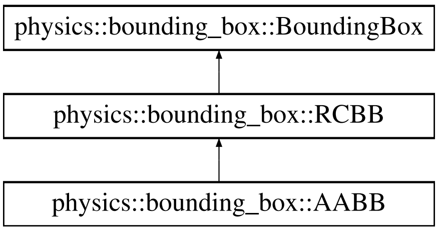
Class Documentation

6.1 physics::bounding_box::AABB Class Reference

BoundingBox aligned on axis.

```
#include <AABB.hpp>
```

Inheritance diagram for physics::bounding_box::AABB:


physics::bounding_box::BoundingBox
physics::bounding_box::RCBB
physics::bounding_box::AABB

Public Member Functions

- AABB ()

  Empty Constructor of an AABB.
- AABB (AABB const &aabb)

  Constructor by copy.
- AABB (glm::vec3 position, glm::vec3 size)

  Constructor of an AABB.
- glm::vec3 get_min ()

  Getter of the minimal value of the AABB.
- glm::vec3 get_max ()

  Getter of the maximal value of the AABB.
- AABB $*$ to_AABB () const override

  Convert the BoundingBox to an AABB.
- void apply_transform (glm::mat4 matrix) override

  Apply the Matrix to the BoundingBox.
- std::vector$<$ glm::vec3 $>$ to_vertices () const override

  Give the vertices that compose the BoundingBox.
- glm::mat3 get_orientation () const override

  Getter of the orientation of the RCBB.
- float get_max_dist ()

  Getter of the max distance between two points in a bb.

Additional Inherited Members

6.1.1 Detailed Description

BoundingBox aligned on axis.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 AABB() [1/3]

```
AABB::AABB ( )
```

Empty Constructor of an AABB.

6.1.2.2 AABB() [2/3]

```
AABB::AABB (
             AABB const & aabb )
```

Constructor by copy.

6.1.2.3 AABB() [3/3]

```
AABB::AABB (
             glm::vec3 position,
             glm::vec3 size )
```

Constructor of an AABB.

6.1.3 Member Function Documentation

6.1.3.1 apply_transform()

```
void AABB::apply_transform (
             glm::mat4 matrix ) [override], [virtual]
```

Apply the Matrix to the BoundingBox.

Parameters

matrix	

Implements physics::bounding_box::BoundingBox.

6.1.3.2 get_max()

```
glm::vec3 AABB::get_max ( )
```

Getter of the maximal value of the AABB.

Returns

max

6.1.3.3 get_max_dist()

```
float AABB::get_max_dist ( ) [virtual]
```

Getter of the max distance between two points in a bb.

Returns

float distance

Reimplemented from physics::bounding_box::RCBB.

6.1.3.4 get_min()

```
glm::vec3 AABB::get_min ( )
```

Getter of the minimal value of the AABB.

Returns

min

6.1.3.5 get_orientation()

```
glm::mat3 AABB::get_orientation ( ) const  [override], [virtual]
```

Getter of the orientation of the RCBB.

Returns

orientation

Implements physics::bounding_box::RCBB.

6.1.3.6 to_AABB()

```
AABB * AABB::to_AABB ( ) const  [override], [virtual]
```

Convert the BoundingBox to an AABB.

Returns

aabb

Implements physics::bounding_box::BoundingBox.

6.1.3.7 to_vertices()

```
std::vector< glm::vec3 > AABB::to_vertices ( ) const  [override], [virtual]
```

Give the vertices that compose the BoundingBox.

Returns

Implements physics::bounding_box::BoundingBox.

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

- src/physics/bounding_box/AABB.hpp
- src/physics/bounding_box/AABB.cpp

6.2 physics::bounding_box::BBFactory Class Reference

Factory of BoundingBox.

```
#include <BBFactory.hpp>
```

Static Public Member Functions

- static BoundingBox ∗ generate_bb (BB_TYPE bb_type)

    Create a BoundingBox according to the type given.

6.2.1 Detailed Description

Factory of BoundingBox.

6.2.2 Member Function Documentation

6.2.2.1 generate_bb()

```
BoundingBox * BBFactory::generate_bb (
              BB_TYPE bb_type )  [static]
```

Create a BoundingBox according to the type given.

Parameters

bb_type	

Returns

bounding box

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

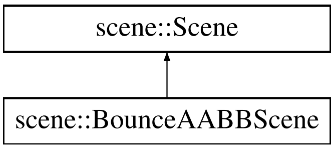
- src/physics/bounding_box/BBFactory.hpp
- src/physics/bounding_box/BBFactory.cpp

6.3 scene::BounceAABBScene Class Reference

Scene with Collisions.

```
#include <BounceAABBScene.hpp>
```

Inheritance diagram for scene::BounceAABBScene:



Public Member Functions

- BounceAABBScene (GLFWwindow *window, const std::string &vertex_shader_path, const std::string &fragment_shader_path, float mult_physics=1.0f)

  Constructor of the BounceAABBScene.

Protected Member Functions

- void process_input (float delta_time) override

  Process the input of the user to have actions on the Scene.

Additional Inherited Members

6.3.1 Detailed Description

Scene with Collisions.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 BounceAABBScene()

```
BounceAABBScene::BounceAABBScene (
            GLFWwindow * window,
            const std::string & vertex_shader_path,
            const std::string & fragment_shader_path,
            float mult_physics = 1.0f )
```

Constructor of the BounceAABBScene.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.3.3 Member Function Documentation

6.3.3.1 process_input()

```
void BounceAABBScene::process_input (
            float delta_time ) [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

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

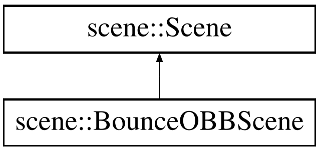
- src/scene/BounceAABBScene.hpp
- src/scene/BounceAABBScene.cpp

6.4 scene::BounceOBBScene Class Reference

Scene with Collisions.

```
#include <BounceOBBScene.hpp>
```

Inheritance diagram for scene::BounceOBBScene:



Public Member Functions

- BounceOBBScene (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_shader_path, float mult_physics=1.0f)

  Constructor of the BounceOBBScene.

Protected Member Functions

- void process_input (float delta_time) override

  Process the input of the user to have actions on the Scene.

Additional Inherited Members

6.4.1 Detailed Description

Scene with Collisions.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 BounceOBBScene()

```
BounceOBBScene::BounceOBBScene (
             GLFWwindow * window,
             const std::string & vertex_shader_path,
             const std::string & fragment_shader_path,
             float mult_physics = 1.0f )
```

Constructor of the BounceOBBScene.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.4.3 Member Function Documentation

6.4.3.1 process_input()

```
void BounceOBBScene::process_input (
             float delta_time )  [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

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

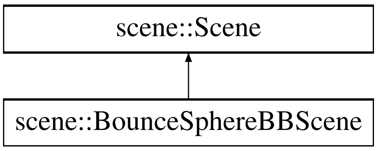
- src/scene/BounceOBBScene.hpp
- src/scene/BounceOBBScene.cpp

6.5 scene::BounceSphereBBScene Class Reference

Scene with Collisions.

```
#include <BounceSphereBBScene.hpp>
```

Inheritance diagram for scene::BounceSphereBBScene:



Public Member Functions

- BounceSphereBBScene (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_shader_path, float mult_physics=1.0f)

    Constructor of the BounceSphereBBScene.

Protected Member Functions

- void process_input (float delta_time) override

    Process the input of the user to have actions on the Scene.

Additional Inherited Members

6.5.1 Detailed Description

Scene with Collisions.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 BounceSphereBBScene()

```
BounceSphereBBScene::BounceSphereBBScene (
            GLFWwindow * window,
            const std::string & vertex_shader_path,
            const std::string & fragment_shader_path,
            float mult_physics = 1.0f )
```

Constructor of the BounceSphereBBScene.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.5.3 Member Function Documentation

6.5.3.1 process_input()

```
void BounceSphereBBScene::process_input (
            float delta_time ) [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

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

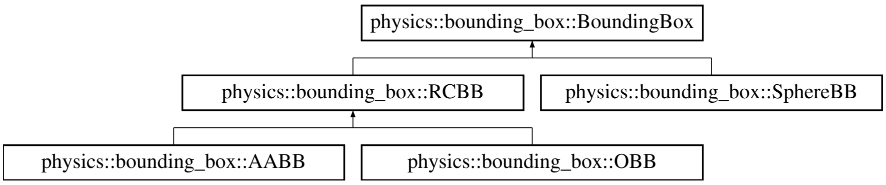
- src/scene/BounceSphereBBScene.hpp
- src/scene/BounceSphereBBScene.cpp

6.6 physics::bounding_box::BoundingBox Class Reference

Represent a bounding box (Abstract)

`#include <BoundingBox.hpp>`

Inheritance diagram for physics::bounding_box::BoundingBox:


physics::bounding_box::BoundingBox
physics::bounding_box::RCBB
physics::bounding_box::SphereBB
physics::bounding_box::AABB
physics::bounding_box::OBB

Public Member Functions

- virtual void compute (std::vector< glm::vec3 > vertices)=0

  Compute the BoundingBox according to vertices.
- bool compute (std::vector< BoundingBox ∗ > bbs)

  Compute the BoundingBox according to a list of BoundingBox.
- BB_TYPE get_type () const

  Getter of the type of the BoundingBox.
- const glm::vec3 & get_position () const

  Getter of the position of the BoundingBox.
- void set_position (const glm::vec3 &position)

Setter of the position of the BoundingBox.

- virtual AABB ∗ to_AABB () const =0

  Convert the BoundingBox to an AABB.
- virtual Collision get_data_collision (SphereBB ∗bb)=0

  Compute the Collision with a SphereBB.
- virtual Collision get_data_collision (RCBB ∗bb)=0

  Compute the Collision with an RCBB.
- virtual void apply_transform (glm::mat4 matrix)=0

  Apply the Matrix to the BoundingBox.
- virtual Interval get_interval (glm::vec3 axis)=0

  Get the Interval clip on an axis.
- virtual std::vector< glm::vec3 > to_vertices () const =0

  Give the vertices that compose the BoundingBox.
- virtual float is_intersected (Ray)=0

  Returns intersection distance along ray direction between ray and boundingbox.
- virtual glm::vec3 closest_point (glm::vec3 pt) const =0

  Returns return closest point on bounding box to pt.
- virtual glm::vec3 get_tensor ()=0

  Getter of the tensor matrix.
- virtual float get_max_dist ()=0

  Getter of the max distance between two points in a bb.

Protected Attributes

- glm::vec3 m_position
- BB_TYPE m_type

6.6.1 Detailed Description

Represent a bounding box (Abstract)

6.6.2 Member Function Documentation

6.6.2.1 apply_transform()

```
virtual void physics::bounding_box::BoundingBox::apply_transform (
            glm::mat4 matrix )  [pure virtual]
```

Apply the Matrix to the BoundingBox.

Parameters

matrix	

Implemented in physics::bounding_box::AABB, physics::bounding_box::SphereBB, and physics::bounding_box::OBB.

6.6.2.2 closest_point()

```
virtual glm::vec3 physics::bounding_box::BoundingBox::closest_point (
            glm::vec3 pt ) const  [pure virtual]
```

Returns return closest point on bounding box to pt.

Returns

Implemented in physics::bounding_box::RCBB, and physics::bounding_box::SphereBB.

6.6.2.3 compute() [1/2]

```
bool BoundingBox::compute (
            std::vector< BoundingBox * > bbs )
```

Compute the BoundingBox according to a list of BoundingBox.

Parameters

bbs	

Returns

has computed

6.6.2.4 compute() [2/2]

```
virtual void physics::bounding_box::BoundingBox::compute (
            std::vector< glm::vec3 > vertices )  [pure virtual]
```

Compute the BoundingBox according to vertices.

Parameters

vertices	

Implemented in physics::bounding_box::RCBB, physics::bounding_box::SphereBB, and physics::bounding_box::OBB.

6.6.2.5 get_data_collision() [1/2]

```
virtual Collision physics::bounding_box::BoundingBox::get_data_collision (
            RCBB * bb )  [pure virtual]
```

Compute the Collision with an RCBB.

Parameters

bb	

Returns

collision

Implemented in physics::bounding_box::RCBB, and physics::bounding_box::SphereBB.

6.6.2.6 get_data_collision() [2/2]

```
virtual Collision physics::bounding_box::BoundingBox::get_data_collision (
            SphereBB * bb )  [pure virtual]
```

Compute the Collision with a SphereBB.

Parameters

bb	

Returns

collision

Implemented in physics::bounding_box::RCBB, and physics::bounding_box::SphereBB.

6.6.2.7 get_interval()

```
virtual Interval physics::bounding_box::BoundingBox::get_interval (
            glm::vec3 axis )  [pure virtual]
```

Get the Interval clip on an axis.

Parameters

axis	

Returns

interval

Implemented in physics::bounding_box::RCBB, and physics::bounding_box::SphereBB.

6.6.2.8 get_max_dist()

```
virtual float physics::bounding_box::BoundingBox::get_max_dist ( ) [pure virtual]
```

Getter of the max distance between two points in a bb.

Returns

float distance

Implemented in physics::bounding_box::SphereBB, physics::bounding_box::OBB, physics::bounding_box::RCBB, and physics::bounding_box::AABB.

6.6.2.9 get_position()

```
const glm::vec3 & BoundingBox::get_position ( ) const
```

Getter of the position of the BoundingBox.

Returns

position

6.6.2.10 get_tensor()

```
virtual glm::vec3 physics::bounding_box::BoundingBox::get_tensor ( ) [pure virtual]
```

Getter of the tensor matrix.

Returns

tensor matrix

Implemented in physics::bounding_box::RCBB, and physics::bounding_box::SphereBB.

6.6.2.11 get_type()

```
BB_TYPE BoundingBox::get_type ( ) const
```

Getter of the type of the BoundingBox.

Returns

type

6.6.2.12 is_intersected()

```
virtual float physics::bounding_box::BoundingBox::is_intersected (
             Ray  )  [pure virtual]
```

Returns intersection distance along ray direction between ray and boundingbox.

Returns

Implemented in physics::bounding_box::SphereBB, and physics::bounding_box::RCBB.

6.6.2.13 set_position()

```
void BoundingBox::set_position (
             const glm::vec3 & position )
```

Setter of the position of the BoundingBox.

Parameters

position	

6.6.2.14 to_AABB()

```
virtual AABB* physics::bounding_box::BoundingBox::to_AABB ( ) const  [pure virtual]
```

Convert the BoundingBox to an AABB.

Returns

aabb

Implemented in physics::bounding_box::RCBB, physics::bounding_box::AABB, and physics::bounding_box::SphereBB.

6.6.2.15 to_vertices()

```
virtual std::vector<glm::vec3> physics::bounding_box::BoundingBox::to_vertices ( ) const
[pure virtual]
```

Give the vertices that compose the BoundingBox.

Returns

Implemented in physics::bounding_box::AABB, physics::bounding_box::SphereBB, and physics::bounding_box::OBB.

6.6.3 Member Data Documentation

6.6.3.1 m_position

```
glm::vec3 physics::bounding_box::BoundingBox::m_position [protected]
```

6.6.3.2 m_type

```
BB_TYPE physics::bounding_box::BoundingBox::m_type [protected]
```

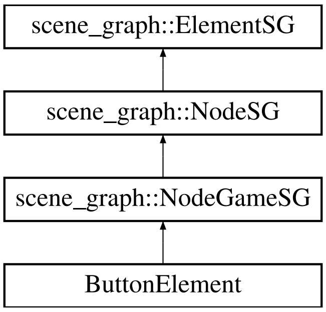
The documentation for this class was generated from the following files:

- src/physics/bounding_box/BoundingBox.hpp
- src/physics/bounding_box/BoundingBox.cpp

6.7 ButtonElement Class Reference

```
#include <ButtonElement.hpp>
```

Inheritance diagram for ButtonElement:



Public Member Functions

- ButtonElement (ElementSG ∗parent, BB_TYPE bb_type, DoorElement ∗=nullptr)

  Constructor of the ButtonElement.
- void add_behavior (float mult, std::vector< RigidBodyVolume ∗ > rbv)

  Add all the RigidBodyBehavior.
- void link_door (DoorElement ∗)

  Link to a door.
- RigidBodyVolume ∗ get_rigid_body ()

  Getter the RigidBodyVolume of the button.

Additional Inherited Members

6.7.1 Constructor & Destructor Documentation

6.7.1.1 ButtonElement()

```
ButtonElement::ButtonElement (
            ElementSG ∗ parent,
            BB_TYPE bb_type,
            DoorElement ∗ door = nullptr )
```

Constructor of the ButtonElement.

Parameters

parent	
bb_type	

6.7.2 Member Function Documentation

6.7.2.1 add_behavior()

```
void ButtonElement::add_behavior (
            float mult,
            std::vector< RigidBodyVolume ∗ > rbv )
```

Add all the RigidBodyBehavior.

Parameters

mult	
rbv	

6.7.2.2 get_rigid_body()

```
RigidBodyVolume * ButtonElement::get_rigid_body ( )
```

Getter the RigidBodyVolume of the button.

Returns

rigid_body

6.7.2.3 link_door()

```
void ButtonElement::link_door (
             DoorElement * door )
```

Link to a door.

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

- src/game_element/ButtonElement.hpp
- src/game_element/ButtonElement.cpp

6.8 Character Class Reference

```
#include <Character.hpp>
```

Public Member Functions

- MouseView * get_mouse_view () const

  Getter of the MouseView.
- void set_mouse_view (MouseView *mouse_view)

  Setter of the MouseView.
- Character (ElementSG *node, PhysicsSystem *ps)

  Constructor of a Character.
- NodeGameSG * get_character_node ()

  Getter of the body node of the Charachter.
- RigidBodyVolume * get_body ()

  Getter of the body.
- NodeGameSG * get_camera ()

  Getter of the camera.
- glm::vec3 get_sight ()

  Getter of the sight of the Character.
- void grab_item (RigidBodyVolume *rbv, double ts, float action_area=12.f)

Grab the RigidBodyVolume given.

- void accumulate_power ()

  Accumulate power to launch item.
- void throw_item (double ts)

  Throw the item handed.
- RigidBodyVolume ∗ get_item ()

  Getter of the item handed.
- bool has_item ()

  Getter of whether or not the Character has an item.
- bool can_interact (double timestamp) const

  Getter of whether or not the Character can interact.
- void jump ()

  Make the Character jump.

6.8.1 Constructor & Destructor Documentation

6.8.1.1 Character()

```
Character::Character (
            ElementSG * node,
            PhysicsSystem * ps )
```

Constructor of a Character.

Parameters

shaders	
node	
ps	

6.8.2 Member Function Documentation

6.8.2.1 accumulate_power()

```
void Character::accumulate_power ( )
```

Accumulate power to launch item.

6.8.2.2 can_interact()

```
bool Character::can_interact (
            double timestamp ) const
```

Getter of whether or not the Character can interact.

Parameters

timestamp	

Returns

can_interact

6.8.2.3 get_body()

```
RigidBodyVolume * Character::get_body ( )
```

Getter of the body.

Returns

body

6.8.2.4 get_camera()

```
NodeGameSG * Character::get_camera ( )
```

Getter of the camera.

6.8.2.5 get_character_node()

```
NodeGameSG * Character::get_character_node ( )
```

Getter of the body node of the Charachter.

Returns

body node

6.8.2.6 get_item()

```
RigidBodyVolume * Character::get_item ( )
```

Getter of the item handed.

Returns

6.8.2.7 get_mouse_view()

```
MouseView * Character::get_mouse_view ( ) const
```

Getter of the MouseView.

Returns

mouse view

6.8.2.8 get_sight()

```
glm::vec3 Character::get_sight ( )
```

Getter of the sight of the Character.

Returns

sight

6.8.2.9 grab_item()

```
void Character::grab_item (
            RigidBodyVolume * rbv,
            double ts,
            float action_area = 12.f )
```

Grab the RigidBodyVolume given.

Parameters

rbv	
ts	
action_area	

6.8.2.10 has_item()

```
bool Character::has_item ( )
```

Getter of whether or not the Character has an item.

Returns

has item

6.8.2.11 jump()

```
void Character::jump ( )
```

Make the Character jump.

6.8.2.12 set_mouse_view()

```
void Character::set_mouse_view (
             MouseView * mouse_view )
```

Setter of the MouseView.

Parameters

mouse_view	

6.8.2.13 throw_item()

```
void Character::throw_item (
             double ts )
```

Throw the item handed.

Parameters

ts	

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

- src/game_element/Character.hpp
- src/game_element/Character.cpp

6.9 physics::Collision Struct Reference

Represents a collision between 2 BoundingBox.

```
#include <Collision.hpp>
```

Public Member Functions

- Collision ()

  Constructor of a Collision.
- void reset ()

  Resets the attributes of a Collision.
- std::string to_string () const

  Give the string that corresponds to the Collision.

Public Attributes

- bool colliding
- glm::vec3 normal
- float depth
- std::vector< glm::vec3 > contacts
- RigidBodyVolume ∗ rigid_body_1 {}
- RigidBodyVolume ∗ rigid_body_2 {}

6.9.1 Detailed Description

Represents a collision between 2 BoundingBox.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 Collision()

```
physics::Collision::Collision ( )  [inline]
```

Constructor of a Collision.

6.9.3 Member Function Documentation

6.9.3.1 reset()

```
void physics::Collision::reset ( )  [inline]
```

Resets the attributes of a Collision.

6.9.3.2 to_string()

```
std::string Collision::to_string ( ) const
```

Give the string that corresponds to the Collision.

Returns

collision as string

6.9.4 Member Data Documentation

6.9.4.1 colliding

```
bool physics::Collision::colliding
```

6.9.4.2 contacts

```
std::vector<glm::vec3> physics::Collision::contacts
```

6.9.4.3 depth

```
float physics::Collision::depth
```

6.9.4.4 normal

```
glm::vec3 physics::Collision::normal
```

6.9.4.5 rigid_body_1

```
RigidBodyVolume* physics::Collision::rigid_body_1 {}
```

6.9.4.6 rigid_body_2

```
RigidBodyVolume* physics::Collision::rigid_body_2 {}
```

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

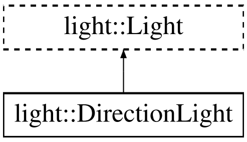
- src/physics/Collision.hpp
- src/physics/Collision.cpp

6.10 light::DirectionLight Class Reference

Light with a direction.

```
#include <DirectionLight.hpp>
```

Inheritance diagram for light::DirectionLight:



Public Member Functions

- DirectionLight (glm::vec3 ambient, glm::vec3 diffuse, glm::vec3 specular)

  Construct a Direction Light (has a DirectionLightBehavior)
- void to_light_info (LightInfo ∗light_struct, glm::mat4 model_mat) override

  fill the data into a LightInfo

Additional Inherited Members

6.10.1 Detailed Description

Light with a direction.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 DirectionLight()

```
DirectionLight::DirectionLight (
            glm::vec3 ambient,
            glm::vec3 diffuse,
            glm::vec3 specular )  [explicit]
```

Construct a Direction Light (has a DirectionLightBehavior)

Parameters

ambient	
diffuse	
specular	

6.10.3 Member Function Documentation

6.10.3.1 to_light_info()

```
void DirectionLight::to_light_info (
            LightInfo * light_shader,
            glm::mat4 model_mat )  [override], [virtual]
```

fill the data into a LightInfo

Parameters

light_shader	
model_mat	

Reimplemented from light::Light.

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

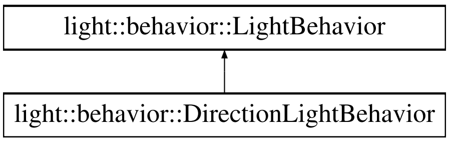
- src/light/DirectionLight.hpp
- src/light/DirectionLight.cpp

6.11 light::behavior::DirectionLightBehavior Class Reference

Behavior of a directed Light.

`#include <DirectionLightBehavior.hpp>`

Inheritance diagram for light::behavior::DirectionLightBehavior:


light::behavior::LightBehavior
light::behavior::DirectionLightBehavior

Public Member Functions

- DirectionLightBehavior ()

  Constructor of a DirectionLightBehavior.
- void apply_to (LightInfo ∗light_shader, glm::mat4 model_mat) override

  Apply the behavior on the LightShader to resolve its data's.

6.11.1 Detailed Description

Behavior of a directed Light.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 DirectionLightBehavior()

```
DirectionLightBehavior::DirectionLightBehavior ( )  [explicit], [default]
```

Constructor of a DirectionLightBehavior.

6.11.3 Member Function Documentation

6.11.3.1 apply_to()

```
void DirectionLightBehavior::apply_to (
            LightInfo * light_info,
            glm::mat4 model_mat )  [override], [virtual]
```

Apply the behavior on the LightShader to resolve its data's.

Parameters

light_info	
model_mat	

Implements light::behavior::LightBehavior.

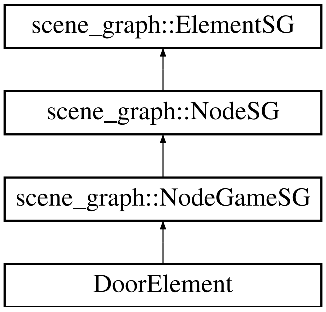
The documentation for this class was generated from the following files:

- src/light/light_behavior/DirectionLightBehavior.hpp
- src/light/light_behavior/DirectionLightBehavior.cpp

6.12 DoorElement Class Reference

```
#include <DoorElement.hpp>
```

Inheritance diagram for DoorElement:



Public Member Functions

- DoorElement (ElementSG ∗parent, BB_TYPE bb_typeint, int id_left_texture, int id_right_texture)

  Constructor of the DoorElement.
- void add_behavior (float mult)

  Add all the RigidBodyBehavior.
- void open ()

  Open the doors.
- void close ()

  Close the doors.
- RigidBodyVolume ∗ get_left_rigid_body ()

  Getter of the left door RigidBodyVolume.
- RigidBodyVolume ∗ get_right_rigid_body ()

  Getter of the right door RigidBodyVolume.

Additional Inherited Members

6.12.1 Constructor & Destructor Documentation

6.12.1.1 DoorElement()

```
DoorElement::DoorElement (
             ElementSG ∗ parent,
             BB_TYPE bb_typeint,
             int id_left_texture,
             int id_right_texture )
```

Constructor of the DoorElement.

Parameters

parent	
bb_typeint	
id_left_texture	
id_right_texture	

6.12.2 Member Function Documentation

6.12.2.1 add_behavior()

```
void DoorElement::add_behavior (
             float mult )
```

Add all the RigidBodyBehavior.

Parameters

mult	

6.12.2.2 close()

```
void DoorElement::close ( )
```

Close the doors.

6.12.2.3 get_left_rigid_body()

```
RigidBodyVolume * DoorElement::get_left_rigid_body ( )
```

Getter of the left door RigidBodyVolume.

Returns

rigid body

6.12.2.4 get_right_rigid_body()

```
RigidBodyVolume * DoorElement::get_right_rigid_body ( )
```

Getter of the right door RigidBodyVolume.

Returns

rigid body

6.12.2.5 open()

```
void DoorElement::open ( )
```

Open the doors.

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

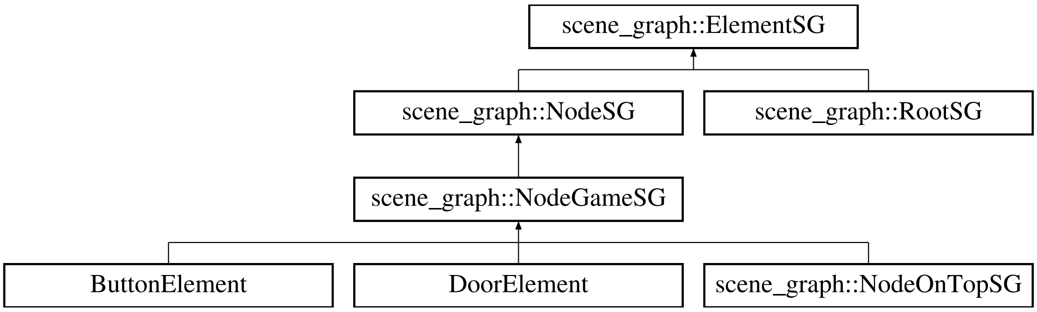
- src/game_element/DoorElement.hpp
- src/game_element/DoorElement.cpp

6.13 scene_graph::ElementSG Class Reference

Base Element of the scene graph (Abstract)

```
#include <ElementSG.hpp>
```

Inheritance diagram for scene_graph::ElementSG:



Public Member Functions

- virtual bool is_node_game ()

  Getter of whether or not the ElementSG is a node of the scene graph that is usable like a node of the game (NodeGameSG)
- void add_child (NodeSG ∗node)

  Add a child to the ElementSG.
- void add_uniform_1i (GLint location, int val=0)

  Add a data to be loaded to the shader at a given location.
- void load_uniforms ()

  Load all the uniform datas to the shader.
- ElementSG ()

  Constructor of the ElementSG.
- Transform ∗ get_trsf ()

  Getter of the transformation matrix of the ElementSG.
- std::vector< NodeSG ∗ > get_children ()

  Getter of all the children NodeSG.
- void remove_child_at (int i)

  Remove a child from the list of NodeSG at a given position.
- void clear_children ()

  Clear the list of NodeSG.
- bool has_children () const

  Getter of whether or not the ElementSG has children.
- virtual glm::mat4 get_matrix_recursive (TransformDirty ∗dirty, bool inverse)=0

  Give the matrix for an extern object (like a child) (recursive function)
- virtual void reset_trsf_dirty (bool dirty)

  Reset all the TransformDirty of the Transform of this node and children recursively.
- virtual void reset_children_dirty (bool dirty)

  Reset the children dirty of the this node and children recursively.
- virtual void compute_trsf_scene_graph ()

  Compute all the Transform list (itself and children)
- virtual void draw (Shaders ∗shaders, glm::vec3 pos_camera, bool allow_debug)

  Draw recursively the graph of the scene.
- virtual ∼ElementSG ()

  Destructor of the ElementSG.

Protected Attributes

- Transform ∗ m_trsf

    Transformation applied to the ElementSG and all the children.
- std::vector< NodeSG ∗ > m_children
- bool m_children_dirty
- std::map< GLuint, int > m_uniform_1i

6.13.1 Detailed Description

Base Element of the scene graph (Abstract)

6.13.2 Constructor & Destructor Documentation

6.13.2.1 ElementSG()

```
ElementSG::ElementSG ( )  [explicit]
```

Constructor of the ElementSG.

6.13.2.2 ∼ElementSG()

```
ElementSG::∼ElementSG ( )  [virtual]
```

Destructor of the ElementSG.

6.13.3 Member Function Documentation

6.13.3.1 add_child()

```
void ElementSG::add_child (
             NodeSG ∗ node )
```

Add a child to the ElementSG.

Parameters

node	

6.13.3.2 add_uniform_1i()

```
void ElementSG::add_uniform_1i (
             GLint location,
             int val = 0 )
```

Add a data to be loaded to the shader at a given location.

Parameters

location	
val	

6.13.3.3 clear_children()

```
void ElementSG::clear_children ( )
```

Clear the list of NodeSG.

6.13.3.4 compute_trsf_scene_graph()

```
void ElementSG::compute_trsf_scene_graph ( )  [virtual]
```

Compute all the Transform list (itself and children)

Reimplemented in scene_graph::NodeSG.

6.13.3.5 draw()

```
void ElementSG::draw (
             Shaders * shaders,
             glm::vec3 pos_camera,
             bool allow_debug )  [virtual]
```

Draw recursively the graph of the scene.

Parameters

shaders	
pos_camera	
allow_debug	

Reimplemented in scene_graph::NodeGameSG, and scene_graph::NodeOnTopSG.

6.13.3.6 get_children()

```
std::vector< NodeSG * > ElementSG::get_children ( )
```

Getter of all the children NodeSG.

Returns

children

6.13.3.7 get_matrix_recursive()

```
virtual glm::mat4 scene_graph::ElementSG::get_matrix_recursive (
            TransformDirty * dirty,
            bool inverse )  [pure virtual]
```

Give the matrix for an extern object (like a child) (recursive function)

Returns

matrix

Implemented in scene_graph::NodeSG, and scene_graph::RootSG.

6.13.3.8 get_trsf()

```
Transform * ElementSG::get_trsf ( )
```

Getter of the transformation matrix of the ElementSG.

Returns

trsf

6.13.3.9 has_children()

```
bool ElementSG::has_children ( ) const
```

Getter of whether or not the ElementSG has children.

Returns

has children

6.13.3.10 is_node_game()

```
bool ElementSG::is_node_game ( )  [virtual]
```

Getter of whether or not the ElementSG is a node of the scene graph that is usable like a node of the game (NodeGameSG)

Returns

is_node_game

Reimplemented in scene_graph::NodeGameSG.

6.13.3.11 load_uniforms()

```
void ElementSG::load_uniforms ( )
```

Load all the uniform datas to the shader.

6.13.3.12 remove_child_at()

```
void ElementSG::remove_child_at (
             int i )
```

Remove a child from the list of NodeSG at a given position.

Parameters

i	

6.13.3.13 reset_children_dirty()

```
void ElementSG::reset_children_dirty (
             bool dirty )  [virtual]
```

Reset the children dirty of the this node and children recursively.

Parameters

dirty	

6.13.3.14 reset_trsf_dirty()

```
void ElementSG::reset_trsf_dirty (
             bool dirty )  [virtual]
```

Reset all the TransformDirty of the Transform of this node and children recursively.

Parameters

dirty	

Reimplemented in scene_graph::NodeSG.

6.13.4 Member Data Documentation

6.13.4.1 m_children

```
std::vector<NodeSG *> scene_graph::ElementSG::m_children  [protected]
```

6.13.4.2 m_children_dirty

```
bool scene_graph::ElementSG::m_children_dirty  [protected]
```

6.13.4.3 m_trsf

```
Transform* scene_graph::ElementSG::m_trsf  [protected]
```

Transformation applied to the ElementSG and all the children.

6.13.4.4 m_uniform_1i

```
std::map<GLuint, int> scene_graph::ElementSG::m_uniform_1i  [protected]
```

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

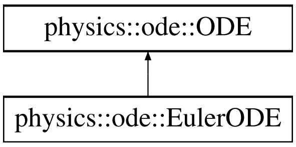
- src/scene_graph/ElementSG.hpp
- src/scene_graph/ElementSG.cpp

6.14 physics::ode::EulerODE Class Reference

Euler Ordinary Differential Equation.

```
#include <EulerODE.hpp>
```

Inheritance diagram for physics::ode::EulerODE:



Public Member Functions

- EulerODE ()
- void update (RigidBodyVolume ∗rbv, float delta_time) override

    Update position and rotation of a RigidBodyVolume with an ODE equation.

Additional Inherited Members

6.14.1 Detailed Description

Euler Ordinary Differential Equation.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 EulerODE()

```
EulerODE::EulerODE ( )
```

6.14.3 Member Function Documentation

6.14.3.1 update()

```
void EulerODE::update (
            RigidBodyVolume ∗ rbv,
            float delta_time )  [override], [virtual]
```

Update position and rotation of a RigidBodyVolume with an ODE equation.

Parameters

rbv	
delta_time	

Implements physics::ode::ODE.

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

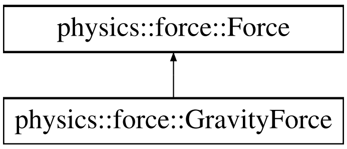
- src/physics/ode/EulerODE.hpp
- src/physics/ode/EulerODE.cpp

6.15 physics::force::Force Class Reference

Represent a Force to apply on a RigidBodyVolume (Abstract)

```
#include <Force.hpp>
```

Inheritance diagram for physics::force::Force:



Public Member Functions

- virtual void apply (RigidBodyVolume ∗rbv)=0

6.15.1 Detailed Description

Represent a Force to apply on a RigidBodyVolume (Abstract)

6.15.2 Member Function Documentation

6.15.2.1 apply()

```
virtual void physics::force::Force::apply (
             RigidBodyVolume * rbv ) [pure virtual]
```

Implemented in physics::force::GravityForce.

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

- src/physics/force/Force.hpp

6.16 shader::glsl_bool Struct Reference

Vec3 used in shader.

```
#include <Shaders.hpp>
```

Public Member Functions

- glsl_bool ()
- glsl_bool (bool boolean)

  Construct a glsl_bool from a boolean.

Public Attributes

- float b

6.16.1 Detailed Description

Vec3 used in shader.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 glsl_bool() [1/2]

```
glsl_bool::glsl_bool ( )
```

6.16.2.2 glsl_bool() [2/2]

```
glsl_bool::glsl_bool (
            bool boolean )  [explicit]
```

Construct a glsl_bool from a boolean.

Parameters

boolean	

6.16.3 Member Data Documentation

6.16.3.1 b

```
float shader::glsl_bool::b
```

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

- src/shader/Shaders.hpp
- src/shader/Shaders.cpp

6.17 shader::glsl_int Struct Reference

Vec3 used in shader.

```
#include <Shaders.hpp>
```

Public Member Functions

- glsl_int ()
- glsl_int (int n)

  Construct a glsl_int from an int.

Public Attributes

- float x

6.17.1 Detailed Description

Vec3 used in shader.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 glsl_int() [1/2]

```
glsl_int::glsl_int ( )
```

6.17.2.2 glsl_int() [2/2]

```
glsl_int::glsl_int (
          int n )  [explicit]
```

Construct a glsl_int from an int.

Parameters

n	

6.17.3 Member Data Documentation

6.17.3.1 x

```
float shader::glsl_int::x
```

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

- src/shader/Shaders.hpp
- src/shader/Shaders.cpp

6.18 shader::glsl_mat4 Struct Reference

Mat4 used in shader.

```
#include <Shaders.hpp>
```

Public Member Functions

- glsl_mat4 ()
- glsl_mat4 (glm::mat4 m)

    Construct a glsl_mat4 from a glm::mat4.

Public Attributes

- float x0
- float x1
- float x2
- float x3
- float y0
- float y1
- float y2
- float y3
- float z0
- float z1
- float z2
- float z3
- float w0
- float w1
- float w2
- float w3

6.18.1 Detailed Description

Mat4 used in shader.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 glsl_mat4() [1/2]

```
glsl_mat4::glsl_mat4 ( )
```

6.18.2.2 glsl_mat4() [2/2]

```
glsl_mat4::glsl_mat4 (
              glm::mat4 m )  [explicit]
```

Construct a glsl_mat4 from a glm::mat4.

Parameters

m	

6.18.3 Member Data Documentation

6.18.3.1 w0

```
float shader::glsl_mat4::w0
```

6.18.3.2 w1

```
float shader::glsl_mat4::w1
```

6.18.3.3 w2

```
float shader::glsl_mat4::w2
```

6.18.3.4 w3

```
float shader::glsl_mat4::w3
```

6.18.3.5 x0

```
float shader::glsl_mat4::x0
```

6.18.3.6 x1

```
float shader::glsl_mat4::x1
```

6.18.3.7 x2

```
float shader::glsl_mat4::x2
```

6.18.3.8 x3

```
float shader::glsl_mat4::x3
```

6.18.3.9 y0

```
float shader::glsl_mat4::y0
```

6.18.3.10 y1

```
float shader::glsl_mat4::y1
```

6.18.3.11 y2

```
float shader::glsl_mat4::y2
```

6.18.3.12 y3

```
float shader::glsl_mat4::y3
```

6.18.3.13 z0

```
float shader::glsl_mat4::z0
```

6.18.3.14 z1

```
float shader::glsl_mat4::z1
```

6.18.3.15 z2

```
float shader::glsl_mat4::z2
```

6.18.3.16 z3

```
float shader::glsl_mat4::z3
```

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

- src/shader/Shaders.hpp
- src/shader/Shaders.cpp

6.19 shader::glsl_vec3 Struct Reference

Vec3 used in shader.

```
#include <Shaders.hpp>
```

Public Member Functions

- glsl_vec3 ()
- glsl_vec3 (glm::vec3 v)

  Construct a glsl_vec3 from a glm::vec3.

Public Attributes

- float x
- float y
- float z

6.19.1 Detailed Description

Vec3 used in shader.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 glsl_vec3() [1/2]

```
glsl_vec3::glsl_vec3 ( )
```

6.19.2.2 glsl_vec3() [2/2]

```
glsl_vec3::glsl_vec3 (
             glm::vec3 v )  [explicit]
```

Construct a glsl_vec3 from a glm::vec3.

Parameters

v	

6.19.3 Member Data Documentation

6.19.3.1 x

```
float shader::glsl_vec3::x
```

6.19.3.2 y

```
float shader::glsl_vec3::y
```

6.19.3.3 z

```
float shader::glsl_vec3::z
```

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

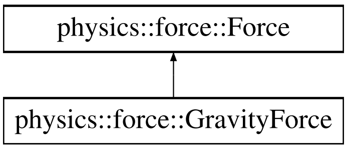
- src/shader/Shaders.hpp
- src/shader/Shaders.cpp

6.20 physics::force::GravityForce Class Reference

Force of the gravity.

```
#include <GravityForce.hpp>
```

Inheritance diagram for physics::force::GravityForce:



Public Member Functions

- GravityForce (glm::vec3 dir={0,-1, 0}, float g=EARTH_GRAVITY)
- void apply (RigidBodyVolume ∗rbv) override

6.20.1 Detailed Description

Force of the gravity.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 GravityForce()

```
GravityForce::GravityForce (
                glm::vec3 dir = {0,-1,0},
                float g = EARTH_GRAVITY )  [explicit]
```

6.20.3 Member Function Documentation

6.20.3.1 apply()

```
void GravityForce::apply (
            RigidBodyVolume * rbv )  [override], [virtual]
```

Implements physics::force::Force.

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

- src/physics/force/GravityForce.hpp
- src/physics/force/GravityForce.cpp

6.21 utils::Interval Struct Reference

Represent an interval between 2 float values.

```
#include <Geometry3D.hpp>
```

Public Member Functions

- Interval (float mi, float ma)

  Constructor of an Interval with a min value and a max value.

Public Attributes

- float min
- float max

6.21.1 Detailed Description

Represent an interval between 2 float values.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 Interval()

```
Interval::Interval (
            float mi,
            float ma )
```

Constructor of an Interval with a min value and a max value.

Parameters

mi	
ma	

6.21.3 Member Data Documentation

6.21.3.1 max

```
float utils::Interval::max
```

6.21.3.2 min

```
float utils::Interval::min
```

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

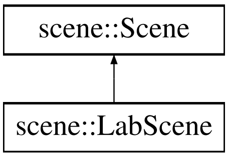
- src/utils/Geometry3D.hpp
- src/utils/Geometry3D.cpp

6.22 scene::LabScene Class Reference

Scene with Collisions.

```
#include <LabScene.hpp>
```

Inheritance diagram for scene::LabScene:



Public Member Functions

- void setRoom (float scale, float mult, int id_tex)
- LabScene (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_↩ shader_path, float mult_physics=1.0f)

  Constructor of the LabScene.
- void update (float delta_time) override

  Render the Scene.
- std::vector< RigidBodyVolume ∗ > get_items ()

  Get all the items grabbable.
- RigidBodyVolume ∗ in_sight ()

  Check whether or not a RigidBodyVolume is grabbable.

Protected Member Functions

- void process_input (float delta_time) override

  Process the input of the user to have actions on the Scene.

Additional Inherited Members

6.22.1 Detailed Description

Scene with Collisions.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 LabScene()

```
LabScene::LabScene (
            GLFWwindow * window,
            const std::string & vertex_shader_path,
            const std::string & fragment_shader_path,
            float mult_physics = 1.0f )
```

Constructor of the LabScene.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.22.3 Member Function Documentation

6.22.3.1 get_items()

```
std::vector< RigidBodyVolume * > LabScene::get_items ( )
```

Get all the items grabbable.

Returns

items

6.22.3.2 in_sight()

```
RigidBodyVolume * LabScene::in_sight ( )
```

Check whether or not a RigidBodyVolume is grabbable.

Returns

in_sight

6.22.3.3 process_input()

```
void LabScene::process_input (
             float delta_time )  [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

6.22.3.4 setRoom()

```
void LabScene::setRoom (
             float scale,
             float mult,
             int id_tex )
```

6.22.3.5 update()

```
void LabScene::update (
             float delta_time )  [override], [virtual]
```

Render the Scene.

Parameters

delta_time	

Reimplemented from scene::Scene.

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

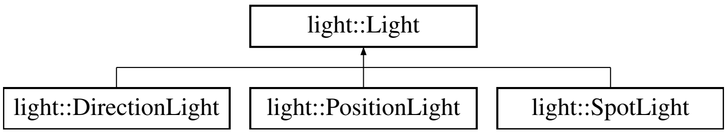
- src/scene/LabScene.hpp
- src/scene/LabScene.cpp

6.23 light::Light Class Reference

Base Light (Abstract)

```
#include <Light.hpp>
```

Inheritance diagram for light::Light:



Public Member Functions

- virtual void to_light_info (LightInfo ∗light_shader, glm::mat4 model_mat)

  fill the data into a LightInfo

Static Public Attributes

- const static int LIGHT_TYPE_DIRECTIONAL = 0
- const static int LIGHT_TYPE_POINT = 1
- const static int LIGHT_TYPE_SPOT = 2

Protected Member Functions

- Light (glm::vec3 ambient, glm::vec3 diffuse, glm::vec3 specular)

  Constructor of the base Light (with ambient, diffuse and specular components) it is protected because we don't want this class to be instantiated.

Protected Attributes

- std::vector< LightBehavior ∗ > m_light_behaviors

6.23.1 Detailed Description

Base Light (Abstract)

6.23.2 Constructor & Destructor Documentation

6.23.2.1 Light()

```
Light::Light (
             glm::vec3 ambient,
             glm::vec3 diffuse,
             glm::vec3 specular )  [explicit], [protected]
```

Constructor of the base Light (with ambient, diffuse and specular components) it is protected because we don't want this class to be instantiated.

Parameters

ambient	
diffuse	
specular	

6.23.3 Member Function Documentation

6.23.3.1 to_light_info()

```
void Light::to_light_info (
             LightInfo * light_shader,
             glm::mat4 model_mat )  [virtual]
```

fill the data into a LightInfo

Parameters

light_shader	
model_mat	

Reimplemented in light::SpotLight, light::DirectionLight, and light::PositionLight.

6.23.4 Member Data Documentation

6.23.4.1 LIGHT_TYPE_DIRECTIONAL

```
const static int light::Light::LIGHT_TYPE_DIRECTIONAL = 0  [static]
```

6.23.4.2 LIGHT_TYPE_POINT

```
const static int light::Light::LIGHT_TYPE_POINT = 1  [static]
```

6.23.4.3 LIGHT_TYPE_SPOT

```
const static int light::Light::LIGHT_TYPE_SPOT = 2  [static]
```

6.23.4.4 m_light_behaviors

```
std::vector<LightBehavior *> light::Light::m_light_behaviors  [protected]
```

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

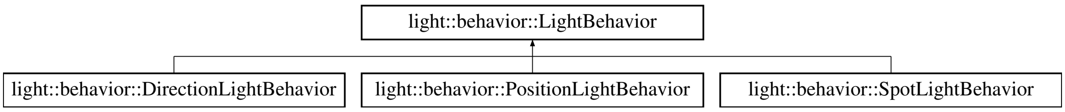
- src/light/Light.hpp
- src/light/Light.cpp

6.24 light::behavior::LightBehavior Class Reference

Behavior of a Light (Abstract)

```
#include <LightBehavior.hpp>
```

Inheritance diagram for light::behavior::LightBehavior:


light::behavior::LightBehavior
light::behavior::DirectionLightBehavior
light::behavior::PositionLightBehavior
light::behavior::SpotLightBehavior

Public Member Functions

- virtual void apply_to (LightInfo ∗light_info, glm::mat4 model_mat)=0

  Apply the behavior on the LightShader to resolve its data's.

6.24.1 Detailed Description

Behavior of a Light (Abstract)

6.24.2 Member Function Documentation

6.24.2.1 apply_to()

```
virtual void light::behavior::LightBehavior::apply_to (
             LightInfo ∗ light_info,
             glm::mat4 model_mat )  [pure virtual]
```

Apply the behavior on the LightShader to resolve its data's.

Parameters

light_info	
model_mat	

Implemented in light::behavior::SpotLightBehavior, light::behavior::PositionLightBehavior, and light::behavior::DirectionLightBehavior.

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

- src/light/light_behavior/LightBehavior.hpp

6.25 light::LightInfo Struct Reference

Light Information.

```
#include <LightShader.hpp>
```

Public Member Functions

- void load_depth_vp_matrix (Shaders ∗shaders)

  Load the depth View and Projection Matrix.

Public Attributes

- int type {}
- glm::vec3 ambient {}
- glm::vec3 diffuse {}
- glm::vec3 specular {}
- glm::vec3 position {}
- float constant_attenuation {}
- float linear_attenuation {}
- float quadratic_attenuation {}
- glm::vec3 direction {}
- float inner_cut_off {}
- float outer_cut_off {}
- bool generate_depth_map {}
- int index_depth_map {}
- glm::mat4 depth_vp_mat
- ShadowMap ∗ shadow_map

6.25.1 Detailed Description

Light Information.

6.25.2 Member Function Documentation

6.25.2.1 load_depth_vp_matrix()

```
void LightInfo::load_depth_vp_matrix (
            Shaders ∗ shaders )
```

Load the depth View and Projection Matrix.

Parameters

shaders	

6.25.3 Member Data Documentation

6.25.3.1 ambient

```
glm::vec3 light::LightInfo::ambient {}
```

6.25.3.2 constant_attenuation

```
float light::LightInfo::constant_attenuation {}
```

6.25.3.3 depth_vp_mat

```
glm::mat4 light::LightInfo::depth_vp_mat
```

6.25.3.4 diffuse

```
glm::vec3 light::LightInfo::diffuse {}
```

6.25.3.5 direction

```
glm::vec3 light::LightInfo::direction {}
```

6.25.3.6 generate_depth_map

```
bool light::LightInfo::generate_depth_map {}
```

6.25.3.7 index_depth_map

```
int light::LightInfo::index_depth_map {}
```

6.25.3.8 inner_cut_off

```
float light::LightInfo::inner_cut_off {}
```

6.25.3.9 linear_attenuation

```
float light::LightInfo::linear_attenuation {}
```

6.25.3.10 outer_cut_off

```
float light::LightInfo::outer_cut_off {}
```

6.25.3.11 position

```
glm::vec3 light::LightInfo::position {}
```

6.25.3.12 quadratic_attenuation

```
float light::LightInfo::quadratic_attenuation {}
```

6.25.3.13 shadow_map

```
ShadowMap* light::LightInfo::shadow_map
```

6.25.3.14 specular

```
glm::vec3 light::LightInfo::specular {}
```

6.25.3.15 type

```
int light::LightInfo::type {}
```

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

- src/light/LightShader.hpp
- src/light/LightShader.cpp

6.26 light::LightShader Struct Reference

Light in the shader based on data retrieved from Light objects

```
#include <LightShader.hpp>
```

Public Member Functions

- LightShader (LightInfo li)

  Constructor of a LightSHader with LightInfo.
- LightShader ()

  Empty COnstrcutor of LightShader.

Public Attributes

- int type {}
- glsl_vec3 ambient {}
- glsl_vec3 diffuse {}
- glsl_vec3 specular {}
- glsl_vec3 position {}
- float constant_attenuation {}
- float linear_attenuation {}
- float quadratic_attenuation {}
- glsl_vec3 direction {}
- float inner_cut_off {}
- float outer_cut_off {}
- int generate_depth_map {}
- int index_sampler_depth_map {}
- glsl_mat4 depth_vp_mat {}

6.26.1 Detailed Description

Light in the shader based on data retrieved from Light objects

6.26.2 Constructor & Destructor Documentation

6.26.2.1 LightShader() [1/2]

```
LightShader::LightShader (
            LightInfo li )  [explicit]
```

Constructor of a LightSHader with LightInfo.

Parameters

li	

6.26.2.2 LightShader() [2/2]

```
LightShader::LightShader ( )  [default]
```

Empty COnstrcutor of LightShader.

6.26.3 Member Data Documentation

6.26.3.1 ambient

```
glsl_vec3 light::LightShader::ambient {}
```

6.26.3.2 constant_attenuation

```
float light::LightShader::constant_attenuation {}
```

6.26.3.3 depth_vp_mat

```
glsl_mat4 light::LightShader::depth_vp_mat {}
```

6.26.3.4 diffuse

```
glsl_vec3 light::LightShader::diffuse {}
```

6.26.3.5 direction

```
glsl_vec3 light::LightShader::direction {}
```

6.26.3.6 generate_depth_map

```
int light::LightShader::generate_depth_map {}
```

6.26.3.7 index_sampler_depth_map

```
int light::LightShader::index_sampler_depth_map {}
```

6.26.3.8 inner_cut_off

```
float light::LightShader::inner_cut_off {}
```

6.26.3.9 linear_attenuation

```
float light::LightShader::linear_attenuation {}
```

6.26.3.10 outer_cut_off

```
float light::LightShader::outer_cut_off {}
```

6.26.3.11 position

```
glsl_vec3 light::LightShader::position {}
```

6.26.3.12 quadratic_attenuation

```
float light::LightShader::quadratic_attenuation {}
```

6.26.3.13 specular

```
glsl_vec3 light::LightShader::specular {}
```

6.26.3.14 type

```
int light::LightShader::type {}
```

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

- src/light/LightShader.hpp
- src/light/LightShader.cpp

6.27 utils::Line Struct Reference

Line with a start point and an end point.

```
#include <Geometry3D.hpp>
```

Public Member Functions

- Line (glm::vec3 s, glm::vec3 e)

  Costructor of a Line with the start point and the end point.
- float length () const

  Getter of the length of the Line.
- bool intersection_plane (Plane plane, glm::vec3 ∗intersection=nullptr) const

  Get the intersection point of the Line on a Plane if the Line intersects the Plane.

Public Attributes

- glm::vec3 start
- glm::vec3 end

6.27.1 Detailed Description

Line with a start point and an end point.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 Line()

```
Line::Line (
            glm::vec3 s,
            glm::vec3 e )
```

Costructor of a Line with the start point and the end point.

Parameters

s	
e	

6.27.3 Member Function Documentation

6.27.3.1 intersection_plane()

```
bool Line::intersection_plane (
            Plane plane,
            glm::vec3 * intersection = nullptr ) const
```

Get the intersection point of the Line on a Plane if the Line intersects the Plane.

Parameters

plane	
intersection	

Returns

6.27.3.2 length()

```
float Line::length ( ) const
```

Getter of the length of the Line.

Returns

length

6.27.4 Member Data Documentation

6.27.4.1 end

```
glm::vec3 utils::Line::end
```

6.27.4.2 start

```
glm::vec3 utils::Line::start
```

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

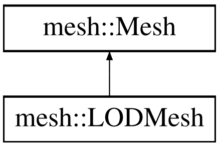
- src/utils/Geometry3D.hpp
- src/utils/Geometry3D.cpp

6.28 mesh::LODMesh Class Reference

Mesh with Level of Details.

```
#include <LODMesh.hpp>
```

Inheritance diagram for mesh::LODMesh:



Public Member Functions

- void update_mesh (float dist_to_camera) override
  Update the datas of the Mesh.
- LODMesh (const MeshData &md, float min_dist, float max_dist, int resol_at_min, int resol_at_max, int levels, BB_TYPE bb_type=AABB_TYPE)
  Constructor of a LODMesh.
- LODMesh (const MeshData &md, float dist_treshold, int resolution, BB_TYPE bb_type=AABB_TYPE)
  Simple constructor of a LODMesh.
- ∼LODMesh ()

Additional Inherited Members

6.28.1 Detailed Description

Mesh with Level of Details.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 LODMesh() [1/2]

```
LODMesh::LODMesh (
                const MeshData & md,
                float min_dist,
                float max_dist,
                int resol_at_min,
                int resol_at_max,
                int levels,
                BB_TYPE bb_type = AABB_TYPE )
```

Constructor of a LODMesh.

Parameters

mesh	base mesh
min_dist	minimum distance of the levels of details
max_dist	maximum distance of the levels of details
resol_at_min	minimum resolution of the simplification of the levels of details
resol_at_max	maximum resolution of the simplification of the levels of details
levels	number of levels of details

6.28.2.2 LODMesh() [2/2]

```
LODMesh::LODMesh (
             const MeshData & md,
             float dist_treshold,
             int resolution,
             BB_TYPE bb_type = AABB_TYPE )
```

Simple constructor of a LODMesh.

Parameters

mesh	base mesh
dist_treshold	treshold of the level of details (only one level here)
resolution	resolution of the simplification of the levels of details

6.28.2.3 ∼LODMesh()

```
LODMesh::∼LODMesh ( )
```

6.28.3 Member Function Documentation

6.28.3.1 update_mesh()

```
void LODMesh::update_mesh (
             float dist_to_camera )  [override], [virtual]
```

Update the datas of the Mesh.

Parameters

dist_to_camera	

Reimplemented from mesh::Mesh.

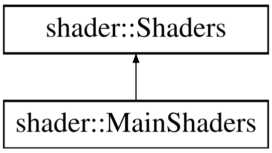
The documentation for this class was generated from the following files:

- src/mesh/LODMesh.hpp
- src/mesh/LODMesh.cpp

6.29 shader::MainShaders Class Reference

```
#include <MainShaders.hpp>
```

Inheritance diagram for shader::MainShaders:



Public Member Functions

- MainShaders (const char ∗vertex_file_path, const char ∗fragment_file_path)

  Constructor of the MainShaders.
- void load_location () const override

  Load all the locations thanks to the ShadersDataManager.
- ShadowMapShaders ∗ get_shadow_map_shaders () const

  Getter of the Shaders of the Shadow Map.

Additional Inherited Members

6.29.1 Constructor & Destructor Documentation

6.29.1.1 MainShaders()

```
MainShaders::MainShaders (
            const char * vertex_file_path,
            const char * fragment_file_path )
```

Constructor of the MainShaders.

Parameters

vertex_file_path	
fragment_file_path	

6.29.2 Member Function Documentation

6.29.2.1 get_shadow_map_shaders()

```
ShadowMapShaders * MainShaders::get_shadow_map_shaders ( ) const
```

Getter of the Shaders of the Shadow Map.

Returns

shadow_map_shaders

6.29.2.2 load_location()

```
void MainShaders::load_location ( ) const  [override], [virtual]
```

Load all the locations thanks to the ShadersDataManager.

Implements shader::Shaders.

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

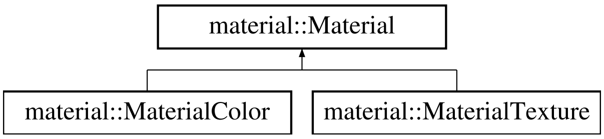
- src/shader/MainShaders.hpp
- src/shader/MainShaders.cpp

6.30 material::Material Class Reference

Base Material (Abstract)

```
#include <Material.hpp>
```

Inheritance diagram for material::Material:



Public Member Functions

- virtual void load_in_shader (Shaders *shaders)

  Load the Material datas in the shader.

Static Public Attributes

- const static int MATERIAL_TYPE_COLOR = 0
- const static int MATERIAL_TYPE_TEXTURE = 1

Protected Member Functions

- Material (float shininess=0.)

  Constructor of a Material.

Protected Attributes

- float m_shininess

6.30.1 Detailed Description

Base Material (Abstract)

6.30.2 Constructor & Destructor Documentation

6.30.2.1 Material()

```
Material::Material (
             float shininess = 0. )  [explicit], [protected]
```

Constructor of a Material.

Parameters

shininess	

6.30.3 Member Function Documentation

6.30.3.1 load_in_shader()

```
void Material::load_in_shader (
             Shaders * shaders )  [virtual]
```

Load the Material datas in the shader.

Parameters

shaders	

Reimplemented in material::MaterialTexture, and material::MaterialColor.

6.30.4 Member Data Documentation

6.30.4.1 m_shininess

```
float material::Material::m_shininess  [protected]
```

6.30.4.2 MATERIAL_TYPE_COLOR

```
const static int material::Material::MATERIAL_TYPE_COLOR = 0  [static]
```

6.30.4.3 MATERIAL_TYPE_TEXTURE

```
const static int material::Material::MATERIAL_TYPE_TEXTURE = 1  [static]
```

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

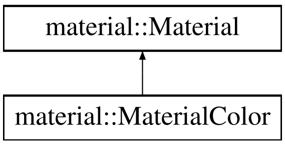
- src/material/Material.hpp
- src/material/Material.cpp

6.31 material::MaterialColor Class Reference

Material with colors for the ambient, diffuse and specular components.

```
#include <MaterialColor.hpp>
```

Inheritance diagram for material::MaterialColor:



Public Member Functions

- MaterialColor (glm::vec3 ambient={0, 0, 0}, glm::vec3 diffuse={0, 0, 0}, glm::vec3 specular={ 0, 0, 0}, float shininess=1.)

  Constructor of a MaterialColor.
- MaterialColor (glm::vec3 color={0, 0, 0}, float shininess=1.)

  Constructor of a MaterialColor that have the same color for the ambient, diffuse and specular components.
- void load_in_shader (Shaders ∗shaders) override

  Load the Material datas in the shader.

Additional Inherited Members

6.31.1 Detailed Description

Material with colors for the ambient, diffuse and specular components.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 MaterialColor() [1/2]

```
MaterialColor::MaterialColor (
             glm::vec3 ambient = {0, 0, 0},
             glm::vec3 diffuse = {0, 0, 0},
             glm::vec3 specular = { 0, 0, 0},
             float shininess = 1. )  [explicit]
```

Constructor of a MaterialColor.

Parameters

ambient	
diffuse	
specular	
shininess	

6.31.2.2 MaterialColor() [2/2]

```
MaterialColor::MaterialColor (
             glm::vec3 color = {0, 0, 0},
             float shininess = 1. )  [explicit]
```

Constructor of a MaterialColor that have the same color for the ambient, diffuse and specular components.

Parameters

color	
shininess	

6.31.3 Member Function Documentation

6.31.3.1 load_in_shader()

```
void MaterialColor::load_in_shader (
            Shaders * shaders )  [override], [virtual]
```

Load the Material datas in the shader.

Parameters

shaders	

Reimplemented from material::Material.

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

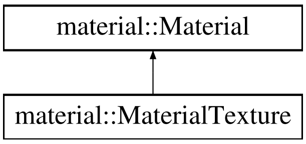
- src/material/MaterialColor.hpp
- src/material/MaterialColor.cpp

6.32 material::MaterialTexture Class Reference

Material with texture for the diffuse and specular components.

```
#include <MaterialTexture.hpp>
```

Inheritance diagram for material::MaterialTexture:



Public Member Functions

- MaterialTexture (TextureManager ∗texture_manager, const std::string &path_diffuse_texture, const std::string &path_specular_texture, float shininess=0.)

  Constructor of a textured Material with the paths to the texture files to has to be loaded.
- MaterialTexture (int id_diffuse_texture, int id_specular_texture, float shininess=0.)

  Constructor of a textured Material with the ids of the texture files already loaded.
- MaterialTexture (int id_texture, float shininess=0.)

  Constructor of a textured Material with the id of the texture file already loaded for the both components diffuse and specular.
- void load_in_shader (Shaders ∗shaders) override

  Load the Material datas in the shader.

Additional Inherited Members

6.32.1 Detailed Description

Material with texture for the diffuse and specular components.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 MaterialTexture() [1/3]

```
MaterialTexture::MaterialTexture (
            TextureManager * texture_manager,
            const std::string & path_diffuse_texture,
            const std::string & path_specular_texture,
            float shininess = 0. )
```

Constructor of a textured Material with the paths to the texture files to has to be loaded.

Parameters

texture_manager	
path_diffuse_texture	
path_specular_texture	
shininess	

6.32.2.2 MaterialTexture() [2/3]

```
MaterialTexture::MaterialTexture (
            int id_diffuse_texture,
            int id_specular_texture,
            float shininess = 0. )
```

Constructor of a textured Material with the ids of the texture files already loaded.

Parameters

id_diffuse_texture	
id_specular_texture	
shininess	

6.32.2.3 MaterialTexture() [3/3]

```
MaterialTexture::MaterialTexture (
             int id_texture,
             float shininess = 0. ) [explicit]
```

Constructor of a textured Material with the id of the texture file already loaded for the both components diffuse and specular.

Parameters

id_texture	
shininess	

6.32.3 Member Function Documentation

6.32.3.1 load_in_shader()

```
void MaterialTexture::load_in_shader (
             Shaders * shaders ) [override], [virtual]
```

Load the Material datas in the shader.

Parameters

shaders	

Reimplemented from material::Material.

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

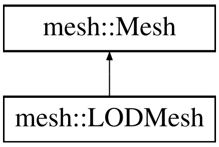
- src/material/MaterialTexture.hpp
- src/material/MaterialTexture.cpp

6.33 mesh::Mesh Class Reference

Base Mesh.

```
#include <Mesh.hpp>
```

Inheritance diagram for mesh::Mesh:



Public Member Functions

- Mesh (const std::vector< glm::vec3 > &vp, const std::vector< unsigned short > &ti, const std::vector< glm::vec2 > &vtc, const std::vector< glm::vec3 > &vn= ∗new std::vector< glm::vec3 >(), bool load_data_now=true, BB_TYPE bb_type=AABB_TYPE)

  Constructor of a Mesh with the vertices position, texture coordinates, normals and triangle indices.
- Mesh (const MeshData &md, bool load_data_now=true, BB_TYPE bb_type=AABB_TYPE)

  Constructor of a Mesh with MeshData.
- Mesh (const char ∗path, bool load_data_now=true, BB_TYPE bb_type=AABB_TYPE)

  Constructor of a Mesh with custom loaded .obj mesh file.
- void load_mesh_in_vao ()

  Load Mesh datas in his VAO.
- virtual void update_mesh (float dist_to_camera)

  Update the datas of the Mesh.
- const std::vector< glm::vec3 > & get_vertex_positions () const

  Getter of the vertex positions.
- const std::vector< unsigned short int > & get_triangle_indices () const

  Getter of the triangle indices.
- const std::vector< glm::vec2 > & get_vertex_tex_coords () const

  Getter of the vertex texture coords.
- const std::vector< glm::vec3 > & get_vertex_normals () const

  Getter of the vertex normals.
- glm::vec3 get_center ()

  Getter of the center of the Mesh.
- bool get_data_at_coords (glm::vec2 pos_in_plan_xz, Transform on_top_trsf, glm::vec3 &position, glm::vec2 &uv, glm::vec3 &normal)

  Gives the data at a given xz position.
- void simplify (int r, float enlargement=0.001)

  Simplify the Mesh with a grid approach (with a resolution) and a Mean Metric Error.
- BoundingBox ∗ get_bb ()

  Getter of the bounding box of the Mesh.
- void load_bb (BB_TYPE bb_type)

  Load the Bounding box of the Mesh.
- GLuint get_vao_id () const

  Getter of the VAO id.
- GLuint get_vbo_position_id () const

  Getter of the VBO of position id.

- GLuint get_vbo_tex_coords_id () const

  Getter of the VBO of texture coords id.
- GLuint get_vbo_normals_id () const

  Getter of the VBO of normals id.
- GLuint get_ebo_triangle_indices_id () const

  Getter of the EBO of triangle indices id.
- ∼Mesh ()

  Destructor of the Mesh.

Protected Attributes

- GLuint m_vbo_position_id {}
- GLuint m_vbo_tex_coords_id {}
- GLuint m_vbo_normals_id {}
- GLuint m_ebo_triangle_indices_id {}
- bool m_loaded_vao
- std::vector< glm::vec3 > m_vertex_positions
- std::vector< unsigned short int > m_triangle_indices
- std::vector< glm::vec2 > m_vertex_tex_coords
- std::vector< glm::vec3 > m_vertex_normals
- BoundingBox ∗ m_bb {}
- glm::vec3 m_center
- GLuint m_vao_id {}

6.33.1 Detailed Description

Base Mesh.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 Mesh() [1/3]

```
Mesh::Mesh (
            const std::vector< glm::vec3 > & vp,
            const std::vector< unsigned short > & ti,
            const std::vector< glm::vec2 > & vtc,
            const std::vector< glm::vec3 > & vn = *new std::vector<glm::vec3>(),
            bool load_data_now = true,
            BB_TYPE bb_type = AABB_TYPE )
```

Constructor of a Mesh with the vertices position, texture coordinates, normals and triangle indices.

Parameters

vp	postions
ti	triangles indices
vtc	vertex texture coordinates
vn	vertex normals

6.33.2.2 Mesh() [2/3]

```
Mesh::Mesh (
            const MeshData & md,
            bool load_data_now = true,
            BB_TYPE bb_type = AABB_TYPE )  [explicit]
```

Constructor of a Mesh with MeshData.

Parameters

md	
load_data_now	
bb_type	

6.33.2.3 Mesh() [3/3]

```
Mesh::Mesh (
            const char * path,
            bool load_data_now = true,
            BB_TYPE bb_type = AABB_TYPE )
```

Constructor of a Mesh with custom loaded .obj mesh file.

Parameters

vp	postions
ti	triangles indices
vtc	vertex texture coordinates
vn	vertex normals

6.33.2.4 ∼Mesh()

```
Mesh::∼Mesh ( )
```

Destructor of the Mesh.

6.33.3 Member Function Documentation

6.33.3.1 get_bb()

```
BoundingBox * Mesh::get_bb ( )
```

Getter of the bounding box of the Mesh.

Parameters

enlargement	

Returns

bb

6.33.3.2 get_center()

```
glm::vec3 Mesh::get_center ( )
```

Getter of the center of the Mesh.

Returns

center

6.33.3.3 get_data_at_coords()

```
bool Mesh::get_data_at_coords (
            glm::vec2 pos_in_plan_xz,
            Transform on_top_trsf,
            glm::vec3 & position,
            glm::vec2 & uv,
            glm::vec3 & normal )
```

Gives the data at a given xz position.

Parameters

pos_in_plan_xz	
on_top_trsf	
position	
uv	
normal	

Returns

data_found

6.33.3.4 get_ebo_triangle_indices_id()

```
GLuint Mesh::get_ebo_triangle_indices_id ( ) const
```

Getter of the EBO of triangle indices id.

Returns

ebo_triangle_indices_id

6.33.3.5 get_triangle_indices()

```
const std::vector< unsigned short int > & Mesh::get_triangle_indices ( ) const
```

Getter of the triangle indices.

Returns

triangle_indices

6.33.3.6 get_vao_id()

```
GLuint Mesh::get_vao_id ( ) const
```

Getter of the VAO id.

Returns

m_vao_id

6.33.3.7 get_vbo_normals_id()

```
GLuint Mesh::get_vbo_normals_id ( ) const
```

Getter of the VBO of normals id.

Returns

vbo_normals_id

6.33.3.8 get_vbo_position_id()

```
GLuint Mesh::get_vbo_position_id ( ) const
```

Getter of the VBO of position id.

Returns

vbo_position_id

6.33.3.9 get_vbo_tex_coords_id()

```
GLuint Mesh::get_vbo_tex_coords_id ( ) const
```

Getter of the VBO of texture coords id.

Returns

vbo_tex_coords_id

6.33.3.10 get_vertex_normals()

```
const std::vector< glm::vec3 > & Mesh::get_vertex_normals ( ) const
```

Getter of the vertex normals.

Returns

vertex_normals

6.33.3.11 get_vertex_positions()

```
const std::vector< glm::vec3 > & Mesh::get_vertex_positions ( ) const
```

Getter of the vertex positions.

Returns

vertex_positions

6.33.3.12 get_vertex_tex_coords()

```
const std::vector< glm::vec2 > & Mesh::get_vertex_tex_coords ( ) const
```

Getter of the vertex texture coords.

Returns

texture_coords

6.33.3.13 load_bb()

```
void Mesh::load_bb (
            BB_TYPE bb_type )
```

Load the Bounding box of the Mesh.

Parameters

bb_type	

6.33.3.14 load_mesh_in_vao()

```
void Mesh::load_mesh_in_vao ( )
```

Load Mesh datas in his VAO.

6.33.3.15 simplify()

```
void Mesh::simplify (
             int r,
             float enlargement = 0.001 )
```

Simplify the Mesh with a grid approach (with a resolution) and a Mean Metric Error.

Parameters

r	
enlargement	

6.33.3.16 update_mesh()

```
void Mesh::update_mesh (
             float dist_to_camera )  [virtual]
```

Update the datas of the Mesh.

Parameters

dist_to_camera	

Reimplemented in mesh::LODMesh.

6.33.4 Member Data Documentation

6.33.4.1 m_bb

```
BoundingBox* mesh::Mesh::m_bb {}  [protected]
```

6.33.4.2 m_center

```
glm::vec3 mesh::Mesh::m_center  [protected]
```

6.33.4.3 m_ebo_triangle_indices_id

```
GLuint mesh::Mesh::m_ebo_triangle_indices_id {}  [protected]
```

6.33.4.4 m_loaded_vao

```
bool mesh::Mesh::m_loaded_vao  [protected]
```

6.33.4.5 m_triangle_indices

```
std::vector<unsigned short int> mesh::Mesh::m_triangle_indices  [protected]
```

6.33.4.6 m_vao_id

```
GLuint mesh::Mesh::m_vao_id {}  [protected]
```

6.33.4.7 m_vbo_normals_id

```
GLuint mesh::Mesh::m_vbo_normals_id {}  [protected]
```

6.33.4.8 m_vbo_position_id

```
GLuint mesh::Mesh::m_vbo_position_id {}  [protected]
```

6.33.4.9 m_vbo_tex_coords_id

```
GLuint mesh::Mesh::m_vbo_tex_coords_id {}  [protected]
```

6.33.4.10 m_vertex_normals

```
std::vector<glm::vec3> mesh::Mesh::m_vertex_normals  [protected]
```

6.33.4.11 m_vertex_positions

```
std::vector<glm::vec3> mesh::Mesh::m_vertex_positions  [protected]
```

6.33.4.12 m_vertex_tex_coords

```
std::vector<glm::vec2> mesh::Mesh::m_vertex_tex_coords  [protected]
```

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

- src/mesh/Mesh.hpp
- src/mesh/Mesh.cpp

6.34 mesh::MeshData Struct Reference

Datas of a Mesh.

```
#include <Mesh.hpp>
```

Public Member Functions

- MeshData (std::vector< glm::vec3 > vp, std::vector< unsigned short int > ti, std::vector< glm::vec2 > vtc, std::vector< glm::vec3 > vn)

  Constructor of MeshData.

Public Attributes

- std::vector< glm::vec3 > vertex_positions
- std::vector< unsigned short int > triangle_indices
- std::vector< glm::vec2 > vertex_tex_coords
- std::vector< glm::vec3 > vertex_normals

6.34.1 Detailed Description

Datas of a Mesh.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 MeshData()

```
MeshData::MeshData (
             std::vector< glm::vec3 > vp,
             std::vector< unsigned short int > ti,
             std::vector< glm::vec2 > vtc,
             std::vector< glm::vec3 > vn )
```

Constructor of MeshData.

Parameters

vp	
ti	
vtc	
vn	

6.34.3 Member Data Documentation

6.34.3.1 triangle_indices

```
std::vector<unsigned short int> mesh::MeshData::triangle_indices
```

6.34.3.2 vertex_normals

```
std::vector<glm::vec3> mesh::MeshData::vertex_normals
```

6.34.3.3 vertex_positions

```
std::vector<glm::vec3> mesh::MeshData::vertex_positions
```

6.34.3.4 vertex_tex_coords

```
std::vector<glm::vec2> mesh::MeshData::vertex_tex_coords
```

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

- src/mesh/Mesh.hpp
- src/utils/meshloader.cpp

6.35 MouseView Class Reference

```
#include <MouseView.hpp>
```

Public Member Functions

- glm::vec3 get_front ()

  Getter of m_front vec3.
- float get_pitch () const

  Getter of the pitch.
- float get_yaw () const

  getter of the yaw

Static Public Member Functions

- static MouseView ∗ get_instance ()

  Getter of the unique instance of this class.
- static void process_mouse (GLFWwindow ∗window, double x, double y)

  Static method called in main mouse callback.

6.35.1 Member Function Documentation

6.35.1.1 get_front()

```
glm::vec3 MouseView::get_front ( )
```

Getter of m_front vec3.

Returns

front

6.35.1.2 get_instance()

```
MouseView * MouseView::get_instance ( )  [static]
```

Getter of the unique instance of this class.

Returns

instance

6.35.1.3 get_pitch()

```
float MouseView::get_pitch ( ) const
```

Getter of the pitch.

Returns

pitch

6.35.1.4 get_yaw()

```
float MouseView::get_yaw ( ) const
```

getter of the yaw

Returns

yaw

6.35.1.5 process_mouse()

```
void MouseView::process_mouse (
             GLFWwindow * window,
             double x,
             double y )  [static]
```

Static method called in main mouse callback.

Parameters

window	
x	
y	

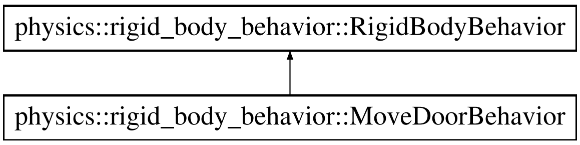
The documentation for this class was generated from the following files:

- src/MouseView.hpp
- src/MouseView.cpp

6.36 physics::rigid_body_behavior::MoveDoorBehavior Class Reference

```
#include <MoveDoorBehavior.hpp>
```

Inheritance diagram for physics::rigid_body_behavior::MoveDoorBehavior:


physics::rigid_body_behavior::RigidBodyBehavior
physics::rigid_body_behavior::MoveDoorBehavior

Public Member Functions

- MoveDoorBehavior (DoorElement ∗door)

  Constructor of MoveDoorBehavior.
- void action (PhysicsSystem ∗ps, Collision collision, float delta_time) override

  Function called each time that the RigidBodyVolume collide.
- void update_physics (float delta_time) override

  Function called each time the physic is updated.
- void update_render (float delta_time, ODE ∗ode) override

  Function called each time the rendering is updated.
- void can_collide_with (RigidBodyVolume ∗rbv)

  Specify a collider that trigger action of the MoveDoorBehavior.

Additional Inherited Members

6.36.1 Constructor & Destructor Documentation

6.36.1.1 MoveDoorBehavior()

```
MoveDoorBehavior::MoveDoorBehavior (
            DoorElement ∗ door )  [explicit]
```

Constructor of MoveDoorBehavior.

Parameters

door	

6.36.2 Member Function Documentation

6.36.2.1 action()

```
void MoveDoorBehavior::action (
             PhysicsSystem * ps,
             Collision collision,
             float delta_time )  [override], [virtual]
```

Function called each time that the RigidBodyVolume collide.

Parameters

ps	
collision	
delta_time	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

6.36.2.2 can_collide_with()

```
void MoveDoorBehavior::can_collide_with (
             RigidBodyVolume * rbv )
```

Specify a collider that trigger action of the MoveDoorBehavior.

Parameters

rbv	

6.36.2.3 update_physics()

```
void MoveDoorBehavior::update_physics (
             float delta_time )  [override], [virtual]
```

Function called each time the physic is updated.

Parameters

delta_time	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

6.36.2.4 update_render()

```
void MoveDoorBehavior::update_render (
            float delta_time,
            ODE * ode )  [override], [virtual]
```

Function called each time the rendering is updated.

Parameters

delta_time	
ode	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

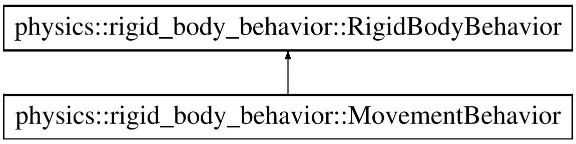
The documentation for this class was generated from the following files:

- src/physics/rigid_body_behavior/MoveDoorBehavior.hpp
- src/physics/rigid_body_behavior/MoveDoorBehavior.cpp

6.37 physics::rigid_body_behavior::MovementBehavior Class Reference

```
#include <MovementBehavior.hpp>
```

Inheritance diagram for physics::rigid_body_behavior::MovementBehavior:


physics::rigid_body_behavior::RigidBodyBehavior
physics::rigid_body_behavior::MovementBehavior

Public Member Functions

- MovementBehavior (bool translatable=true, bool rotatable=true, float mult_physics=1.0f, float mass=1.0f, float friction=0.6f, float cor=0.5f)
- void action (PhysicsSystem ∗ps, Collision collision, float delta_time) override

  Function called each time that the RigidBodyVolume collide.
- void update_physics (float delta_time) override

  Function called each time the physic is updated.
- void update_render (float delta_time, ODE ∗ode) override

  Function called each time the rendering is updated.
- bool is_translatable () const

  Getter of whether or not the RigidBodyVolume is translatable.
- bool is_rotatable () const

  Getter of whether or not the RigidBodyVolume is rotatable.

- void add_force (Force ∗f)

    Add a force to the list of force to apply at each updates.
- float inverse_mass () const

    Getter of the inverse of the mass.
- glm::mat4 inverse_tensor () const

    Getter of the inverse of the tensor.
- void add_linear_impulse (glm::vec3 &impulse)

    Add a linear impulse to the RigidBodyVolume.
- void add_rotational_impulse (glm::vec3 &point, glm::vec3 &impulse)

    Add a rotational impulse to the RigidBodyVolume.
- void apply_impulse (RigidBodyVolume &rbv, const Collision &collision, int index_contact)

    Apply an impulse to the RigidBodyVolume.
- float get_mass () const

    Getter of the mass.
- void set_mass (float)

    Setter of the mass.
- void set_forces (const glm::vec3 &forces)

    Setter of the forces.
- void clear_forces ()

    Clear the list of forces.
- glm::vec3 get_forces () const

    Getter of the forces.
- glm::vec3 get_torques () const

    Getter of the torques.
- glm::vec3 get_velocity () const

    Getter of the velocity.
- glm::vec3 get_acceleration () const

    Getter of the velocity.
- void set_velocity (const glm::vec3 &velocity)

    Setter of the velocity.
- void set_angular_velocity (const glm::vec3 &angular_velocity)

    Setter of the angular velocity.
- void set_acceleration (const glm::vec3 &acceleration)

    Setter of the acceleration.
- glm::vec3 get_angular_velocity () const

    Getter of the angular velocity.
- glm::vec3 get_angular_acceleration () const

    Getter of the angular acceleration.

Additional Inherited Members

6.37.1 Constructor & Destructor Documentation

6.37.1.1 MovementBehavior()

```
MovementBehavior::MovementBehavior (
             bool translatable = true,
             bool rotatable = true,
             float mult_physics = 1.0f,
             float mass = 1.0f,
             float friction = 0.6f,
             float cor = 0.5f )  [explicit]
```

6.37.2 Member Function Documentation

6.37.2.1 action()

```
void MovementBehavior::action (
             PhysicsSystem * ps,
             Collision collision,
             float delta_time )  [override], [virtual]
```

Function called each time that the RigidBodyVolume collide.

Parameters

ps	
collision	
delta_time	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

6.37.2.2 add_force()

```
void MovementBehavior::add_force (
             Force * f )
```

Add a force to the list of force to apply at each updates.

Parameters

f	

6.37.2.3 add_linear_impulse()

```
void MovementBehavior::add_linear_impulse (
            glm::vec3 & impulse )
```

Add a linear impulse to the RigidBodyVolume.

Parameters

impulse	

6.37.2.4 add_rotational_impulse()

```
void MovementBehavior::add_rotational_impulse (
            glm::vec3 & point,
            glm::vec3 & impulse )
```

Add a rotational impulse to the RigidBodyVolume.

Parameters

point	
impulse	

6.37.2.5 apply_impulse()

```
void MovementBehavior::apply_impulse (
            RigidBodyVolume & rbv,
            const Collision & collision,
            int index_contact )
```

Apply an impulse to the RigidBodyVolume.

Parameters

rbv	
collision	

6.37.2.6 clear_forces()

```
void MovementBehavior::clear_forces ( )
```

Clear the list of forces.

6.37.2.7 get_acceleration()

```
glm::vec3 MovementBehavior::get_acceleration ( ) const
```

Getter of the velocity.

Returns

velocity

6.37.2.8 get_angular_acceleration()

```
glm::vec3 MovementBehavior::get_angular_acceleration ( ) const
```

Getter of the angular acceleration.

Returns

angular acceleration

6.37.2.9 get_angular_velocity()

```
glm::vec3 MovementBehavior::get_angular_velocity ( ) const
```

Getter of the angular velocity.

Returns

angular velocity

6.37.2.10 get_forces()

```
glm::vec3 MovementBehavior::get_forces ( ) const
```

Getter of the forces.

Returns

forces

6.37.2.11 get_mass()

```
float MovementBehavior::get_mass ( ) const
```

Getter of the mass.

Returns

mass

6.37.2.12 get_torques()

```
glm::vec3 MovementBehavior::get_torques ( ) const
```

Getter of the torques.

Returns

torques

6.37.2.13 get_velocity()

```
glm::vec3 MovementBehavior::get_velocity ( ) const
```

Getter of the velocity.

Returns

velocity

6.37.2.14 inverse_mass()

```
float MovementBehavior::inverse_mass ( ) const
```

Getter of the inverse of the mass.

Returns

inverse mass

6.37.2.15 inverse_tensor()

```
glm::mat4 MovementBehavior::inverse_tensor ( ) const
```

Getter of the inverse of the tensor.

Returns

inverse tensor

6.37.2.16 is_rotatable()

```
bool MovementBehavior::is_rotatable ( ) const
```

Getter of whether or not the RigidBodyVolume is rotatable.

Returns

is rotatable

6.37.2.17 is_translatable()

```
bool MovementBehavior::is_translatable ( ) const
```

Getter of whether or not the RigidBodyVolume is translatable.

Returns

is translatable

6.37.2.18 set_acceleration()

```
void MovementBehavior::set_acceleration (
            const glm::vec3 & acceleration )
```

Setter of the acceleration.

Parameters

acceleration	

6.37.2.19 set_angular_velocity()

```
void MovementBehavior::set_angular_velocity (
             const glm::vec3 & angular_velocity )
```

Setter of the angular velocity.

Parameters

angular_velocity	

6.37.2.20 set_forces()

```
void MovementBehavior::set_forces (
             const glm::vec3 & forces )
```

Setter of the forces.

Parameters

forces	

6.37.2.21 set_mass()

```
void MovementBehavior::set_mass (
             float mass )
```

Setter of the mass.

Returns

mass

6.37.2.22 set_velocity()

```
void MovementBehavior::set_velocity (
             const glm::vec3 & velocity )
```

Setter of the velocity.

Parameters

velocity	

6.37.2.23 update_physics()

```
void MovementBehavior::update_physics (
            float delta_time )  [override], [virtual]
```

Function called each time the physic is updated.

Parameters

delta_time	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

6.37.2.24 update_render()

```
void MovementBehavior::update_render (
            float delta_time,
            ODE * ode )  [override], [virtual]
```

Function called each time the rendering is updated.

Parameters

delta_time	
ode	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

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

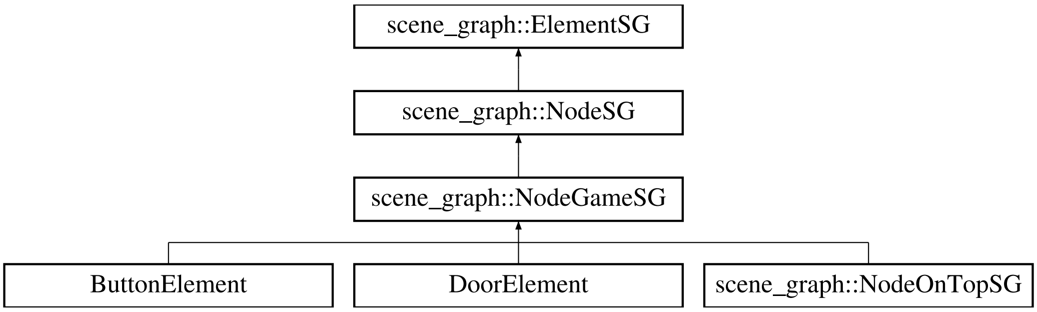
- src/physics/rigid_body_behavior/MovementBehavior.hpp
- src/physics/rigid_body_behavior/MovementBehavior.cpp

6.38 scene_graph::NodeGameSG Class Reference

NodeSG that can have Meshes, light and camera.

```
#include <NodeGameSG.hpp>
```

Inheritance diagram for scene_graph::NodeGameSG:



Public Member Functions

- NodeGameSG (ElementSG ∗parent, BB_TYPE bb_type=AABB_TYPE)

    Constructor of the NodeGameSG.
- void set_debug_rendering (bool dr, glm::vec3 color_rendering={1, 0, 0})

    Setter of the Debug rendering.
- bool is_node_game () override

    Getter of whether or not the ElementSG is a node of the scene graph that is usable like a node of the game (NodeGameSG)
- RigidBodyVolume ∗ get_rigid_body () const

    Getter of the RigidBodyVolume.
- void set_rigid_body (RigidBodyVolume ∗rigid_body)

    Setter of the RigidBodyVolume.
- bool refresh_bb (glm::vec3 pos_camera, bool change_dirty_flags=true)

    Refresh the BoundingBox of the NodeGameSG.
- void set_light (Light ∗light)

    Set a light at this NodeGameSG.
- void remove_light ()

    Remove the light to this NodeGameSG.
- bool has_light ()

    Getter of if the NodeGameSG has a light or no.
- LightInfo generate_light_struct ()

    Give the LightInfo that correspond to the Light.
- void update_view_mat (Shaders ∗shaders)

    Update the view matrix in the shader.
- void update_view_pos (Shaders ∗shaders)

    pdate the view position in the shader
- void draw (Shaders ∗shaders, glm::vec3 pos_camera, bool allow_debug) override

    Draw recursively the graph of the scene.
- void set_meshes (std::vector< Mesh ∗ > meshes)

    Setter of the Mesh list of the NodeGameSG.
- void set_material (Material ∗material)

    Setter of the material applied to the NodeGameSG.
- Material ∗ get_material () const

    Getter of the material.
- const std::vector< Mesh ∗ > & get_meshes () const

    Getter of the Mesh list of the NodeGameSG.
- bool has_meshes () const

    Getter of whether or not the ElementSG has children.
- glm::vec3 get_center (glm::vec3 pos_camera)

Gives the center of the NodeGameSG (mean of each Mesh centers)

- void set_see_both_face (bool see_both_face)

  Setter of the flag see_both_face (resolve both face of the Meshes))
- std::pair< glm::vec3, glm::vec3 > get_aabb (glm::vec3 pos_camera)

  Gives the aligned axis bounding box of the NodeGameSG (bounding box of Mesh bounding boxes)
- float get_distance_from (glm::vec3 cam_position, glm::vec3 position)

  Compute the distance between the bounding box and a given position.
- BoundingBox * get_bb ()

  Getter of the BoundingBox.
- void set_drawable (bool drawable)

  Setter of whether or not the NodeGameSG is drable.

Additional Inherited Members

6.38.1 Detailed Description

NodeSG that can have Meshes, light and camera.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 NodeGameSG()

```
NodeGameSG::NodeGameSG (
            ElementSG * parent,
            BB_TYPE bb_type = AABB_TYPE )
```

Constructor of the NodeGameSG.

Parameters

parent	
bb_type	

6.38.3 Member Function Documentation

6.38.3.1 draw()

```
void NodeGameSG::draw (
            Shaders * shaders,
            glm::vec3 pos_camera,
            bool allow_debug )  [override], [virtual]
```

Draw recursively the graph of the scene.

Parameters

shaders	
pos_camera	
allow_debug	

Reimplemented from scene_graph::ElementSG.

Reimplemented in scene_graph::NodeOnTopSG.

6.38.3.2 generate_light_struct()

```
LightInfo NodeGameSG::generate_light_struct ( )
```

Give the LightInfo that correspond to the Light.

Returns

light_shader

6.38.3.3 get_aabb()

```
std::pair< glm::vec3, glm::vec3 > NodeGameSG::get_aabb (
             glm::vec3 pos_camera )
```

Gives the aligned axis bounding box of the NodeGameSG (bounding box of Mesh bounding boxes)

Parameters

pos_camera	

Returns

bounding_box

6.38.3.4 get_bb()

```
BoundingBox * NodeGameSG::get_bb ( )
```

Getter of the BoundingBox.

Returns

bb

6.38.3.5 get_center()

```
glm::vec3 NodeGameSG::get_center (
            glm::vec3 pos_camera )
```

Gives the center of the NodeGameSG (mean of each Mesh centers)

Parameters

pos_camera	

Returns

center

6.38.3.6 get_distance_from()

```
float NodeGameSG::get_distance_from (
            glm::vec3 cam_position,
            glm::vec3 position )
```

Compute the distance between the bounding box and a given position.

Parameters

cam_position	
position	

Returns

distance

6.38.3.7 get_material()

```
Material * NodeGameSG::get_material ( ) const
```

Getter of the material.

Returns

material

6.38.3.8 get_meshes()

```
const std::vector< Mesh * > & NodeGameSG::get_meshes ( ) const
```

Getter of the Mesh list of the NodeGameSG.

Returns

meshes

6.38.3.9 get_rigid_body()

```
RigidBodyVolume * NodeGameSG::get_rigid_body ( ) const
```

Getter of the RigidBodyVolume.

Returns

rigid_body

6.38.3.10 has_light()

```
bool NodeGameSG::has_light ( )
```

Getter of if the NodeGameSG has a light or no.

Returns

has light

6.38.3.11 has_meshes()

```
bool NodeGameSG::has_meshes ( ) const
```

Getter of whether or not the ElementSG has children.

Returns

has meshes

6.38.3.12 is_node_game()

```
bool NodeGameSG::is_node_game ( )  [override], [virtual]
```

Getter of whether or not the ElementSG is a node of the scene graph that is usable like a node of the game (NodeGameSG)

Returns

is_node_game

Reimplemented from scene_graph::ElementSG.

6.38.3.13 refresh_bb()

```
bool NodeGameSG::refresh_bb (
            glm::vec3 pos_camera,
            bool change_dirty_flags = true )
```

Refresh the BoundingBox of the NodeGameSG.

Parameters

pos_camera	
change_dirty_flags	

Returns

has refresh

6.38.3.14 remove_light()

```
void NodeGameSG::remove_light ( )
```

Remove the light to this NodeGameSG.

6.38.3.15 set_debug_rendering()

```
void NodeGameSG::set_debug_rendering (
            bool dr,
            glm::vec3 color_rendering = {1, 0, 0} )
```

Setter of the Debug rendering.

Parameters

dr	
color_rendering	

6.38.3.16 set_drawable()

```
void NodeGameSG::set_drawable (
             bool drawable )
```

Setter of whether or not the NodeGameSG is drable.

Parameters

drawable	

6.38.3.17 set_light()

```
void NodeGameSG::set_light (
             Light * light )
```

Set a light at this NodeGameSG.

Parameters

light	

6.38.3.18 set_material()

```
void NodeGameSG::set_material (
             Material * material )
```

Setter of the material applied to the NodeGameSG.

Parameters

material	

6.38.3.19 set_meshes()

```
void NodeGameSG::set_meshes (
            std::vector< Mesh * > meshes )
```

Setter of the Mesh list of the NodeGameSG.

Parameters

meshes	

6.38.3.20 set_rigid_body()

```
void NodeGameSG::set_rigid_body (
            RigidBodyVolume * rigid_body )
```

Setter of the RigidBodyVolume.

Parameters

rigid_body	

6.38.3.21 set_see_both_face()

```
void NodeGameSG::set_see_both_face (
            bool see_both_face )
```

Setter of the flag see_both_face (resolve both face of the Meshes))

Parameters

see_both_face	

6.38.3.22 update_view_mat()

```
void NodeGameSG::update_view_mat (
            Shaders * shaders )
```

Update the view matrix in the shader.

Parameters

shaders	

6.38.3.23 update_view_pos()

```
void NodeGameSG::update_view_pos (
            Shaders * shaders )
```

pdate the view position in the shader

Parameters

shaders	

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

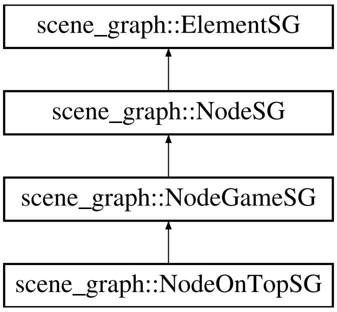
- src/scene_graph/NodeGameSG.hpp
- src/scene_graph/NodeGameSG.cpp

6.39 scene_graph::NodeOnTopSG Class Reference

NodeGameSG on top of another.

`#include <NodeOnTopSG.hpp>`

Inheritance diagram for scene_graph::NodeOnTopSG:



Public Member Functions

- NodeOnTopSG (ElementSG ∗parent, NodeGameSG ∗on_node, BB_TYPE bb_type=AABB_TYPE)

  Constructor of NodeOnTopSG.
- bool get_data_on (glm::vec3 pos_camera, glm::vec3 &position, glm::vec2 &uv, glm::vec3 &normal, float &height_adjustement)

  Compute the data according to the NodeGameSG under.
- void draw (Shaders ∗shaders, glm::vec3 pos_camera, bool allow_debug) override

  Draw recursively the graph of the scene.

Additional Inherited Members

6.39.1 Detailed Description

NodeGameSG on top of another.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 NodeOnTopSG()

```
NodeOnTopSG::NodeOnTopSG (
            ElementSG * parent,
            NodeGameSG * on_node,
            BB_TYPE bb_type = AABB_TYPE )
```

Constructor of NodeOnTopSG.

Parameters

parent	
on_node	
bb_type	

6.39.3 Member Function Documentation

6.39.3.1 draw()

```
void NodeOnTopSG::draw (
            Shaders * shaders,
            glm::vec3 pos_camera,
            bool allow_debug )  [override], [virtual]
```

Draw recursively the graph of the scene.

Parameters

shaders	
pos_camera	
allow_debug	

Reimplemented from scene_graph::NodeGameSG.

6.39.3.2 get_data_on()

```
bool NodeOnTopSG::get_data_on (
            glm::vec3 pos_camera,
            glm::vec3 & position,
            glm::vec2 & uv,
            glm::vec3 & normal,
            float & height_adjustement )
```

Compute the data according to the NodeGameSG under.

Parameters

pos_camera	
position	
uv	
normal	
height_adjustement	

Returns

found

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

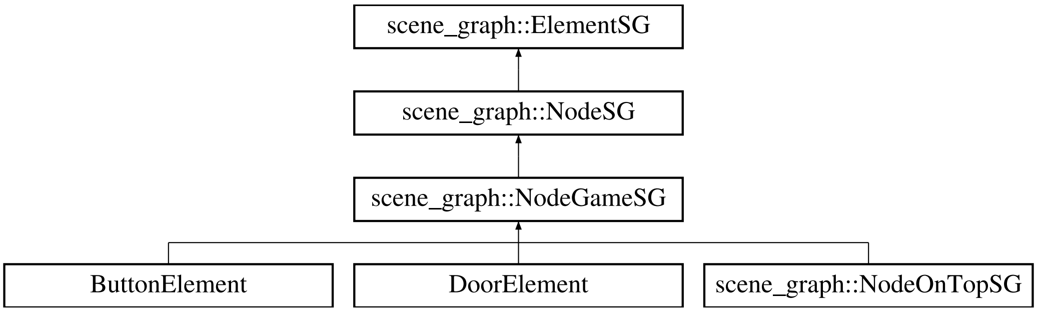
- src/scene_graph/NodeOnTopSG.hpp
- src/scene_graph/NodeOnTopSG.cpp

6.40 scene_graph::NodeSG Class Reference

Base Node of the scene graph (Abstract)

```
#include <NodeSG.hpp>
```

Inheritance diagram for scene_graph::NodeSG:



Public Member Functions

- void load_model_matrices (Shaders ∗shaders)

  Load the Model matrices of the NodeSG in the Shaders.
- Transform ∗ get_local_trsf ()

  Getter of the local transformation matrix of the NodeSG.
- glm::mat4 get_matrix_recursive (TransformDirty ∗dirty, bool inverse) override

  Give the matrix for an extern object (like a child) (recursive function)
- glm::mat4 get_matrix_recursive_local (TransformDirty ∗dirty=nullptr, bool inverse=false)

  Give the matrix for the NodeSG (recursive function)
- void compute_trsf_scene_graph () override

  Compute all the Transform list (itself and children)
- glm::vec3 get_position_in_world (glm::vec3 v={0, 0, 0})

  Compute the position in the world of the NodeSG.
- void set_parent (ElementSG ∗)
- void reset_trsf_dirty (bool dirty) override

  Reset all the TransformDirty of the Transform of this node and children recursively.
- ∼NodeSG () override

Protected Member Functions

- NodeSG (ElementSG ∗parent)

  Constructor of the NodeSG.

Protected Attributes

- Transform ∗ m_local_trsf

6.40.1 Detailed Description

Base Node of the scene graph (Abstract)

6.40.2 Constructor & Destructor Documentation

6.40.2.1 NodeSG()

```
NodeSG::NodeSG (
            ElementSG * parent )  [protected]
```

Constructor of the NodeSG.

Parameters

parent	

6.40.2.2 ∼NodeSG()

```
NodeSG::∼NodeSG ( )  [override]
```

6.40.3 Member Function Documentation

6.40.3.1 compute_trsf_scene_graph()

```
void NodeSG::compute_trsf_scene_graph ( )  [override], [virtual]
```

Compute all the Transform list (itself and children)

Reimplemented from scene_graph::ElementSG.

6.40.3.2 get_local_trsf()

```
Transform * NodeSG::get_local_trsf ( )
```

Getter of the local transformation matrix of the NodeSG.

Returns

trsf

6.40.3.3 get_matrix_recursive()

```
glm::mat4 NodeSG::get_matrix_recursive (
             TransformDirty * dirty,
             bool inverse )  [override], [virtual]
```

Give the matrix for an extern object (like a child) (recursive function)

Returns

matrix

Implements scene_graph::ElementSG.

6.40.3.4 get_matrix_recursive_local()

```
glm::mat4 NodeSG::get_matrix_recursive_local (
             TransformDirty * dirty = nullptr,
             bool inverse = false )
```

Give the matrix for the NodeSG (recursive function)

Returns

matrix

6.40.3.5 get_position_in_world()

```
glm::vec3 NodeSG::get_position_in_world (
             glm::vec3 v = {0, 0, 0} )
```

Compute the position in the world of the NodeSG.

Parameters

v	

Returns

position

6.40.3.6 load_model_matrices()

```
void NodeSG::load_model_matrices (
             Shaders * shaders )
```

Load the Model matrices of the NodeSG in the Shaders.

Parameters

shaders	

6.40.3.7 reset_trsf_dirty()

```
void NodeSG::reset_trsf_dirty (
             bool dirty ) [override], [virtual]
```

Reset all the TransformDirty of the Transform of this node and children recursively.

Parameters

dirty	

Reimplemented from scene_graph::ElementSG.

6.40.3.8 set_parent()

```
void NodeSG::set_parent (
            ElementSG * parent )
```

6.40.4 Member Data Documentation

6.40.4.1 m_local_trsf

```
Transform* scene_graph::NodeSG::m_local_trsf [protected]
```

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

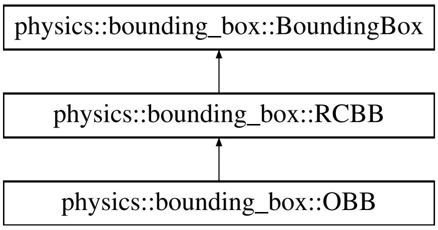
- src/scene_graph/NodeSG.hpp
- src/scene_graph/NodeSG.cpp

6.41 physics::bounding_box::OBB Class Reference

BoundingBox oriented.

```
#include <OBB.hpp>
```

Inheritance diagram for physics::bounding_box::OBB:


physics::bounding_box::BoundingBox
physics::bounding_box::RCBB
physics::bounding_box::OBB

Public Member Functions

- OBB ()

  Constructor of an OBB.
- void compute (std::vector< glm::vec3 > vertices) override

  Compute the BoundingBox according to vertices.
- glm::mat3 get_orientation () const override

  Getter of the orientation of the RCBB.
- std::vector< glm::vec3 > to_vertices () const override

  Give the vertices that compose the BoundingBox.
- void apply_transform (glm::mat4 matrix) override

  Apply the Matrix to the BoundingBox.
- float get_max_dist () override

  Getter of the max distance between two points in a bb.

Additional Inherited Members

6.41.1 Detailed Description

BoundingBox oriented.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 OBB()

```
OBB::OBB ( )
```

Constructor of an OBB.

6.41.3 Member Function Documentation

6.41.3.1 apply_transform()

```
void OBB::apply_transform (
            glm::mat4 matrix )  [override], [virtual]
```

Apply the Matrix to the BoundingBox.

Parameters

matrix	

Implements physics::bounding_box::BoundingBox.

6.41.3.2 compute()

```
void OBB::compute (
             std::vector< glm::vec3 > vertices )  [override], [virtual]
```

Compute the BoundingBox according to vertices.

Parameters

vertices	

Implements physics::bounding_box::BoundingBox.

6.41.3.3 get_max_dist()

```
float OBB::get_max_dist ( )  [override], [virtual]
```

Getter of the max distance between two points in a bb.

Returns

float distance

Reimplemented from physics::bounding_box::RCBB.

6.41.3.4 get_orientation()

```
glm::mat3 OBB::get_orientation ( ) const  [override], [virtual]
```

Getter of the orientation of the RCBB.

Returns

orientation

Implements physics::bounding_box::RCBB.

6.41.3.5 to_vertices()

```
std::vector< glm::vec3 > OBB::to_vertices ( ) const  [override], [virtual]
```

Give the vertices that compose the BoundingBox.

Returns

Implements physics::bounding_box::BoundingBox.

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

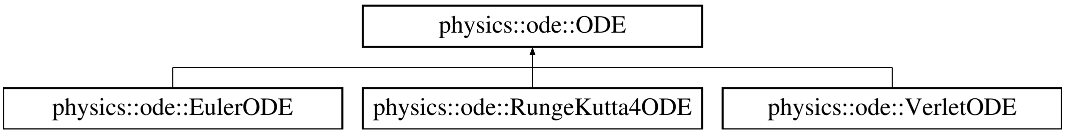
- src/physics/bounding_box/OBB.hpp
- src/physics/bounding_box/OBB.cpp

6.42 physics::ode::ODE Class Reference

Ordinary Differential Equation (Abstract)

```
#include <ODE.hpp>
```

Inheritance diagram for physics::ode::ODE:



Public Member Functions

- virtual void update (RigidBodyVolume ∗rbv, float delta_time)=0

  Update position and rotation of a RigidBodyVolume with an ODE equation.
- ODE_TYPE get_type () const

  Getter of the type of the ODE.

Protected Attributes

- ODE_TYPE m_type {}

6.42.1 Detailed Description

Ordinary Differential Equation (Abstract)

6.42.2 Member Function Documentation

6.42.2.1 get_type()

```
ODE_TYPE ODE::get_type ( ) const
```

Getter of the type of the ODE.

Returns

type

6.42.2.2 update()

```
virtual void physics::ode::ODE::update (
            RigidBodyVolume * rbv,
            float delta_time ) [pure virtual]
```

Update position and rotation of a RigidBodyVolume with an ODE equation.

Parameters

rbv	
delta_time	

Implemented in physics::ode::EulerODE, physics::ode::RungeKutta4ODE, and physics::ode::VerletODE.

6.42.3 Member Data Documentation

6.42.3.1 m_type

```
ODE_TYPE physics::ode::ODE::m_type {} [protected]
```

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

- src/physics/ode/ODE.hpp
- src/physics/ode/ODE.cpp

6.43 physics::ode::ODEFactory Class Reference

Factory of ODE.

```
#include <ODEFactory.hpp>
```

Static Public Member Functions

- static ODE ∗ generate_ode (ODE_TYPE ode_type)

  Create an ODE according to the type given.

6.43.1 Detailed Description

Factory of ODE.

6.43.2 Member Function Documentation

6.43.2.1 generate_ode()

```
ODE * ODEFactory::generate_ode (
            ode::ODE_TYPE ode_type )  [static]
```

Create an ODE according to the type given.

Parameters

ode_type	

Returns

ode

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

- src/physics/ode/ODEFactory.hpp
- src/physics/ode/ODEFactory.cpp

6.44 physics::PhysicsSystem Class Reference

Manage a whole physic system with RigidBody.

```
#include <PhysicsSystem.hpp>
```

Public Member Functions

- PhysicsSystem (ElementSG ∗root_physics, float mult_physics=1.0f, float lpp=0.2f, float ps=0.01f, int ii=5, ODE_TYPE ode_type=EULER_TYPE)

  Constructor of PhysicsSystem.
- void add_collider (RigidBodyVolume ∗rbv)

  Add a RigidBodyVolume to the PhysicsSystem.
- void remove_collider (RigidBodyVolume ∗rbv)

  Remove a RigidBodyVolume to the PhysicsSystem.
- void clear_rigid_bodies ()

  Clear all the RigidBodyVolume.
- void remove_collider_with_node (NodeGameSG ∗node)

  Clear the RigidBodyVolume with the NodeGameSG given.
- void update_collisions (glm::vec3 pos_camera, float delta_time)

  Update the Collisions in the PhysicsSystem.
- void update_bodies (glm::vec3 pos_camera, float delta_time)

  Update the RigidBodyVolume in the PhysicsSystem.
- void refresh_bodies_bb (glm::vec3 pos_camera)

  Refresh the BoundingBox of each RigidBodyVolume.
- int get_impulse_iteration () const

  Getter of the impulse iteration.
- float get_penetration_slack () const

  Getter of the penetration slack.
- float get_linear_projection_percent () const

  Getter of the linear projection percent.
- void set_multiplicator_physics (float mult_physics)

  Setter of the Physics forces multiplicator.
- float get_multiplicator_physics () const

  Getter of the Physics forces multiplicator.

6.44.1 Detailed Description

Manage a whole physic system with RigidBody.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 PhysicsSystem()

```
PhysicsSystem::PhysicsSystem (
            ElementSG ∗ root_physics,
            float mult_physics = 1.0f,
            float lpp = 0.2f,
            float ps = 0.01f,
            int ii = 5,
            ODE_TYPE ode_type = EULER_TYPE )  [explicit]
```

Constructor of PhysicsSystem.

Parameters

lpp	
ps	
ii	

6.44.3 Member Function Documentation

6.44.3.1 add_collider()

```
void PhysicsSystem::add_collider (
             physics::RigidBodyVolume * rbv )
```

Add a RigidBodyVolume to the PhysicsSystem.

Parameters

rbv	

6.44.3.2 clear_rigid_bodies()

```
void PhysicsSystem::clear_rigid_bodies ( )
```

Clear all the RigidBodyVolume.

6.44.3.3 get_impulse_iteration()

```
int PhysicsSystem::get_impulse_iteration ( ) const
```

Getter of the impulse iteration.

Returns

impulse iteration

6.44.3.4 get_linear_projection_percent()

```
float PhysicsSystem::get_linear_projection_percent ( ) const
```

Getter of the linear projection percent.

Returns

linear projection percent

6.44.3.5 get_multiplicator_physics()

```
float PhysicsSystem::get_multiplicator_physics ( ) const
```

Getter of the Physics forces multiplicator.

Returns

multiplicator

6.44.3.6 get_penetration_slack()

```
float PhysicsSystem::get_penetration_slack ( ) const
```

Getter of the penetration slack.

Returns

penetration slack

6.44.3.7 refresh_bodies_bb()

```
void PhysicsSystem::refresh_bodies_bb (
             glm::vec3 pos_camera )
```

Refresh the BoundingBox of each RigidBodyVolume.

Parameters

pos_camera	

6.44.3.8 remove_collider()

```
void PhysicsSystem::remove_collider (
             physics::RigidBodyVolume * rbv )
```

Remove a RigidBodyVolume to the PhysicsSystem.

Parameters

rbv	

6.44.3.9 remove_collider_with_node()

```
void PhysicsSystem::remove_collider_with_node (
             NodeGameSG * node )
```

Clear the RigidBodyVolume with the NodeGameSG given.

Parameters

node	

6.44.3.10 set_multiplicator_physics()

```
void PhysicsSystem::set_multiplicator_physics (
             float mult_physics )
```

Setter of the Physics forces multiplicator.

Parameters

mult_physics	

6.44.3.11 update_bodies()

```
void PhysicsSystem::update_bodies (
             glm::vec3 pos_camera,
             float delta_time )
```

Update the RigidBodyVolume in the PhysicsSystem.

Parameters

pos_camera	
delta_time	

6.44.3.12 update_collisions()

```
void PhysicsSystem::update_collisions (
            glm::vec3 pos_camera,
            float delta_time )
```

Update the Collisions in the PhysicsSystem.

Parameters

pos_camera	
delta_time	

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

- src/physics/PhysicsSystem.hpp
- src/physics/PhysicsSystem.cpp

6.45 utils::Plane Struct Reference

Plane with a normal and a distance from the origin along the axis of the normal.

```
#include <Geometry3D.hpp>
```

Public Member Functions

- Plane (glm::vec3 n, float d)

  Constructor of a Plane with the normal and the distance from the origin.
- float equation (glm::vec3 point) const

  Inform if a point is on the Plane (=0), behind the Plane (<0) or in front of the Plane (>0)

Static Public Member Functions

- static Plane plane_from_normal_point (glm::vec3 n, glm::vec3 p)

  Constructor of a Plane with a normal and a point on the Plane.
- static Plane plane_from_normal_vectors (glm::vec3 p, glm::vec3 v1, glm::vec3 v2)

  Constructor of a Plane with a point on the Plane and 2 vectors non-collinear on the plane.
- static Plane plane_from_points (glm::vec3 v1, glm::vec3 v2, glm::vec3 v3)

  Constructor of a Plane with a 3 points on the Plane.

Public Attributes

- glm::vec3 normal
- float distance

  Distance from origin.

6.45.1 Detailed Description

Plane with a normal and a distance from the origin along the axis of the normal.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 Plane()

```
Plane::Plane (
            glm::vec3 n,
            float d )
```

Constructor of a Plane with the normal and the distance from the origin.

Parameters

n	
d	

6.45.3 Member Function Documentation

6.45.3.1 equation()

```
float Plane::equation (
            glm::vec3 point ) const
```

Inform if a point is on the Plane (=0), behind the Plane (<0) or in front of the Plane (>0)

Parameters

point	

Returns

result of the equation

6.45.3.2 plane_from_normal_point()

```
Plane Plane::plane_from_normal_point (
             glm::vec3 n,
             glm::vec3 p )  [static]
```

Constructor of a Plane with a normal and a point on the Plane.

Parameters

n	
p	

6.45.3.3 plane_from_normal_vectors()

```
Plane Plane::plane_from_normal_vectors (
             glm::vec3 p,
             glm::vec3 v1,
             glm::vec3 v2 )  [static]
```

Constructor of a Plane with a point on the Plane and 2 vectors non-collinear on the plane.

Parameters

p	
v1	
v2	

6.45.3.4 plane_from_points()

```
Plane Plane::plane_from_points (
             glm::vec3 v1,
             glm::vec3 v2,
             glm::vec3 v3 )  [static]
```

Constructor of a Plane with a 3 points on the Plane.

Parameters

v1	
v2	
v3	

Returns

6.45.4 Member Data Documentation

6.45.4.1 distance

```
float utils::Plane::distance
```

Distance from origin.

6.45.4.2 normal

```
glm::vec3 utils::Plane::normal
```

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

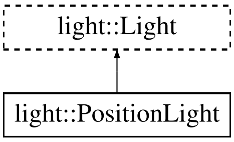
- src/utils/Geometry3D.hpp
- src/utils/Geometry3D.cpp

6.46 light::PositionLight Class Reference

Light with a position and whose intensity can be attenuated.

```
#include <PositionLight.hpp>
```

Inheritance diagram for light::PositionLight:



Public Member Functions

- PositionLight (glm::vec3 ambient, glm::vec3 diffuse, glm::vec3 specular, float constant_attenuation=0.f, float linear_attenuation=0.f, float quadratic_attenuation=0.f)

  Constructor of a Positionned Light (has a PositionLightBehavior)
- void to_light_info (LightInfo ∗light_shader, glm::mat4 model_mat) override

  fill the data into a LightInfo

Additional Inherited Members

6.46.1 Detailed Description

Light with a position and whose intensity can be attenuated.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 PositionLight()

```
PositionLight::PositionLight (
             glm::vec3 ambient,
             glm::vec3 diffuse,
             glm::vec3 specular,
             float constant_attenuation = 0.f,
             float linear_attenuation = 0.f,
             float quadratic_attenuation = 0.f )  [explicit]
```

Constructor of a Positionned Light (has a PositionLightBehavior)

Parameters

ambient	
diffuse	
specular	
constant_attenuation	
linear_attenuation	
quadratic_attenuation	

6.46.3 Member Function Documentation

6.46.3.1 to_light_info()

```
void PositionLight::to_light_info (
             LightInfo * light_shader,
             glm::mat4 model_mat )  [override], [virtual]
```

fill the data into a LightInfo

Parameters

light_shader	
model_mat	

Reimplemented from light::Light.

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

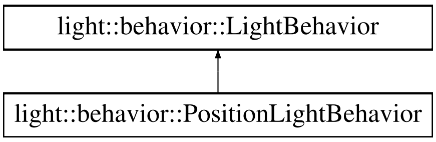
- src/light/PositionLight.hpp
- src/light/PositionLight.cpp

6.47 light::behavior::PositionLightBehavior Class Reference

Behavior of a positionned Light.

```
#include <PositionLightBehavior.hpp>
```

Inheritance diagram for light::behavior::PositionLightBehavior:


light::behavior::LightBehavior
light::behavior::PositionLightBehavior

Public Member Functions

- PositionLightBehavior (float ca, float la, float qa)

  Constructor if a PositionLightBehavior with the 3 attenuations.
- void apply_to (LightInfo ∗light_shader, glm::mat4 model_mat) override

  Apply the behavior on the LightShader to resolve its data's.

6.47.1 Detailed Description

Behavior of a positionned Light.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 PositionLightBehavior()

```
PositionLightBehavior::PositionLightBehavior (
              float ca,
              float la,
              float qa )  [explicit]
```

Constructor if a PositionLightBehavior with the 3 attenuations.

Parameters

ca	
la	
qa	

6.47.3 Member Function Documentation

6.47.3.1 apply_to()

```
void PositionLightBehavior::apply_to (
            LightInfo * light_info,
            glm::mat4 model_mat )  [override], [virtual]
```

Apply the behavior on the LightShader to resolve its data's.

Parameters

light_info	
model_mat	

Implements light::behavior::LightBehavior.

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

- src/light/light_behavior/PositionLightBehavior.hpp
- src/light/light_behavior/PositionLightBehavior.cpp

6.48 utils::Ray Struct Reference

Ray with an origine and a direction.

```
#include <Geometry3D.hpp>
```

Public Member Functions

- Ray (glm::vec3 o, glm::vec3 d)

  Constructor of a Ray with the origin and the direction.

Public Attributes

- glm::vec3 origin
- glm::vec3 direction

6.48.1 Detailed Description

Ray with an origine and a direction.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 Ray()

```
Ray::Ray (
            glm::vec3 o,
            glm::vec3 d )
```

Constructor of a Ray with the origin and the direction.

Parameters

o	
d	

6.48.3 Member Data Documentation

6.48.3.1 direction

```
glm::vec3 utils::Ray::direction
```

6.48.3.2 origin

```
glm::vec3 utils::Ray::origin
```

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

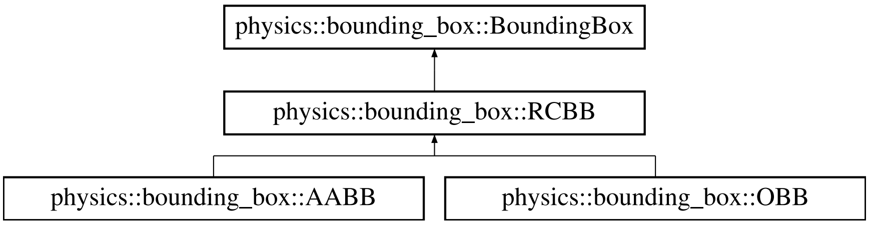
- src/utils/Geometry3D.hpp
- src/utils/Geometry3D.cpp

6.49 physics::bounding_box::RCBB Class Reference

Rectangle Cuboid BoundingBox (Abstract)

```
#include <RCBB.hpp>
```

Inheritance diagram for physics::bounding_box::RCBB:


physics::bounding_box::BoundingBox
physics::bounding_box::RCBB
physics::bounding_box::AABB
physics::bounding_box::OBB

Public Member Functions

- void compute (std::vector< glm::vec3 > vertices) override

  Compute the BoundingBox according to vertices.
- std::vector< Line > to_edges () const

  Give the edges that compose the RCBB.
- std::vector< Plane > to_planes () const

  Give the planes that compose the RCBB.
- std::vector< glm::vec3 > get_intersections_lines (std::vector< Line > lines) const

  Get all the intersections with a list of Line.
- float penetrate_depth (RCBB *bb, glm::vec3 axis, bool *out_should_flip)

  Compute the depth of the penetration between 2 RCBB colliding.
- virtual glm::mat3 get_orientation () const =0

  Getter of the orientation of the RCBB.
- bool is_point_in (glm::vec3 point) const

  Check if a point is in the RCBB.
- AABB * to_AABB () const override

  Convert the BoundingBox to an AABB.
- Interval get_interval (glm::vec3 axis) override

  Get the Interval clip on an axis.
- glm::vec3 get_tensor () override

  Getter of the tensor matrix.
- Collision get_data_collision (RCBB *bb) override

  Compute the Collision with an RCBB.
- Collision get_data_collision (SphereBB *bb) override

  Compute the Collision with a SphereBB.
- glm::vec3 closest_point (glm::vec3 pt) const override

  Returns return closest point on bounding box to pt.
- float is_intersected (Ray ray) override

  Returns intersection distance along ray direction between ray and boundingbox.
- virtual float get_max_dist ()

  Getter of the max distance between two points in a bb.

Protected Attributes

- glm::vec3 m_size

6.49.1 Detailed Description

Rectangle Cuboid BoundingBox (Abstract)

6.49.2 Member Function Documentation

6.49.2.1 closest_point()

```
glm::vec3 RCBB::closest_point (
             glm::vec3 pt ) const  [override], [virtual]
```

Returns return closest point on bounding box to pt.

Returns

Implements physics::bounding_box::BoundingBox.

6.49.2.2 compute()

```
void RCBB::compute (
             std::vector< glm::vec3 > vertices )  [override], [virtual]
```

Compute the BoundingBox according to vertices.

Parameters

vertices	

Implements physics::bounding_box::BoundingBox.

6.49.2.3 get_data_collision() [1/2]

```
Collision RCBB::get_data_collision (
             RCBB * bb )  [override], [virtual]
```

Compute the Collision with an RCBB.

Parameters

bb	

Returns

collision

Implements physics::bounding_box::BoundingBox.

6.49.2.4 get_data_collision() [2/2]

```
Collision RCBB::get_data_collision (
            SphereBB * bb ) [override], [virtual]
```

Compute the Collision with a SphereBB.

Parameters

bb	

Returns

collision

Implements physics::bounding_box::BoundingBox.

6.49.2.5 get_intersections_lines()

```
std::vector< glm::vec3 > RCBB::get_intersections_lines (
            std::vector< Line > lines ) const
```

Get all the intersections with a list of Line.

Parameters

edges	

Returns

intersections

6.49.2.6 get_interval()

```
Interval RCBB::get_interval (
             glm::vec3 axis )  [override], [virtual]
```

Get the Interval clip on an axis.

Parameters

axis	

Returns

interval

Implements physics::bounding_box::BoundingBox.

6.49.2.7 get_max_dist()

```
float RCBB::get_max_dist ( )  [virtual]
```

Getter of the max distance between two points in a bb.

Returns

float distance

Implements physics::bounding_box::BoundingBox.

Reimplemented in physics::bounding_box::OBB, and physics::bounding_box::AABB.

6.49.2.8 get_orientation()

```
virtual glm::mat3 physics::bounding_box::RCBB::get_orientation ( ) const  [pure virtual]
```

Getter of the orientation of the RCBB.

Returns

orientation

Implemented in physics::bounding_box::AABB, and physics::bounding_box::OBB.

6.49.2.9 get_tensor()

```
glm::vec3 RCBB::get_tensor ( )  [override], [virtual]
```

Getter of the tensor matrix.

Returns

tensor matrix

Implements physics::bounding_box::BoundingBox.

6.49.2.10 is_intersected()

```
float RCBB::is_intersected (
             Ray  )  [override], [virtual]
```

Returns intersection distance along ray direction between ray and boundingbox.

Returns

Implements physics::bounding_box::BoundingBox.

6.49.2.11 is_point_in()

```
bool RCBB::is_point_in (
             glm::vec3 point ) const
```

Check if a point is in the RCBB.

Parameters

point	

Returns

is point in

6.49.2.12 penetrate_depth()

```
float RCBB::penetrate_depth (
             RCBB * bb,
             glm::vec3 axis,
             bool * out_should_flip )
```

Compute the depth of the penetration between 2 RCBB colliding.

Parameters

bb	
axis	
out_should_flip	

Returns

depth

6.49.2.13 to_AABB()

```
AABB * RCBB::to_AABB ( ) const [override], [virtual]
```

Convert the BoundingBox to an AABB.

Returns

aabb

Implements physics::bounding_box::BoundingBox.

6.49.2.14 to_edges()

```
std::vector< Line > RCBB::to_edges ( ) const
```

Give the edges that compose the RCBB.

Returns

6.49.2.15 to_planes()

```
std::vector< Plane > RCBB::to_planes ( ) const
```

Give the planes that compose the RCBB.

Returns

6.49.3 Member Data Documentation

6.49.3.1 m_size

```
glm::vec3 physics::bounding_box::RCBB::m_size  [protected]
```

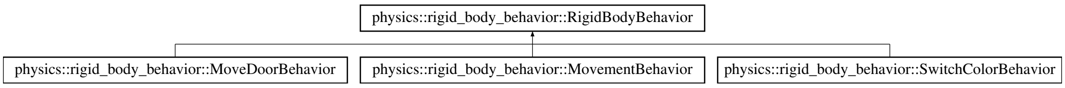
The documentation for this class was generated from the following files:

- src/physics/bounding_box/RCBB.hpp
- src/physics/bounding_box/RCBB.cpp

6.50 physics::rigid_body_behavior::RigidBodyBehavior Class Reference

```
#include <RigidBodyBehavior.hpp>
```

Inheritance diagram for physics::rigid_body_behavior::RigidBodyBehavior:


physics::rigid_body_behavior::RigidBodyBehavior
physics::rigid_body_behavior::MoveDoorBehavior
physics::rigid_body_behavior::MovementBehavior
physics::rigid_body_behavior::SwitchColorBehavior

Public Member Functions

- virtual void action (PhysicsSystem ∗ps, Collision collision, float delta_time)=0

    Function called each time that the RigidBodyVolume collide.
- virtual void update_physics (float delta_time)=0

    Function called each time the physic is updated.
- virtual void update_render (float delta_time, ODE ∗ode)=0

    Function called each time the rendering is updated.
- void set_rigid_body (RigidBodyVolume ∗rbv)

    Setter of the RigidBodyVolume.
- RigidBodyBehavior_TYPE get_type () const

    Getter of the type.

Protected Attributes

- RigidBodyVolume ∗ m_rigid_body {}
- RigidBodyBehavior_TYPE m_type {}

6.50.1 Member Function Documentation

6.50.1.1 action()

```
virtual void physics::rigid_body_behavior::RigidBodyBehavior::action (
            PhysicsSystem ∗ ps,
            Collision collision,
            float delta_time )  [pure virtual]
```

Function called each time that the RigidBodyVolume collide.

Parameters

ps	
collision	
delta_time	

Implemented in physics::rigid_body_behavior::MovementBehavior, physics::rigid_body_behavior::SwitchColorBehavior, and physics::rigid_body_behavior::MoveDoorBehavior.

6.50.1.2 get_type()

```
rigid_body_behavior::RigidBodyBehavior_TYPE rigid_body_behavior::RigidBodyBehavior::get_type (
) const
```

Getter of the type.

Returns

type

6.50.1.3 set_rigid_body()

```
void RigidBodyBehavior::set_rigid_body (
            RigidBodyVolume * rbv )
```

Setter of the RigidBodyVolume.

Parameters

rbv	

6.50.1.4 update_physics()

```
virtual void physics::rigid_body_behavior::RigidBodyBehavior::update_physics (
            float delta_time )  [pure virtual]
```

Function called each time the physic is updated.

Parameters

delta_time	

Implemented in physics::rigid_body_behavior::MovementBehavior, physics::rigid_body_behavior::SwitchColorBehavior, and physics::rigid_body_behavior::MoveDoorBehavior.

6.50.1.5 update_render()

```
virtual void physics::rigid_body_behavior::RigidBodyBehavior::update_render (
            float delta_time,
            ODE * ode )  [pure virtual]
```

Function called each time the rendering is updated.

Parameters

delta_time	
ode	

Implemented in physics::rigid_body_behavior::MovementBehavior, physics::rigid_body_behavior::SwitchColorBehavior, and physics::rigid_body_behavior::MoveDoorBehavior.

6.50.2 Member Data Documentation

6.50.2.1 m_rigid_body

```
RigidBodyVolume* physics::rigid_body_behavior::RigidBodyBehavior::m_rigid_body {}  [protected]
```

6.50.2.2 m_type

```
RigidBodyBehavior_TYPE physics::rigid_body_behavior::RigidBodyBehavior::m_type {}  [protected]
```

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

- src/physics/rigid_body_behavior/RigidBodyBehavior.hpp
- src/physics/rigid_body_behavior/RigidBodyBehavior.cpp

6.51 physics::RigidBodyVolume Class Reference

Represents a rigid body with a volume (BoundingBox)

```
#include <RigidBodyVolume.hpp>
```

Public Member Functions

- RigidBodyVolume (NodeGameSG ∗ng, bool character=false)

  Constructor of a RigidBodyVolume with a NodeGameSG and configurable coefficients.
- ∼RigidBodyVolume ()

  Destructor of a RigidBodyVolume.
- Collision find_data_collision (RigidBodyVolume &rbv)

  Compute if there is a Collision between this RigidBodyVolume and another.
- NodeGameSG ∗ get_node ()

  Getter of the NodeGameSG.
- void add_behavior (RigidBodyBehavior ∗behavior)

  Add a RigidBodyBehavior to the RigidBodyVolume.
- bool has_movement_behavior ()

  Getter of whether or not the node has a MovementBehavior.
- MovementBehavior ∗ get_movement_behavior ()

  Getter of the MovementBehavior.
- void action (PhysicsSystem ∗ps, const Collision &collision, float delta_time)

  Call the action function of each RigidBodyBehavior of the RigidBodyVolume.
- void update_physics (float delta_time)

  Call the update_physics function of each RigidBodyBehavior of the RigidBodyVolume.
- void update_render (float delta_time, ODE ∗ode)

  Call the update_render function of each RigidBodyBehavior of the RigidBodyVolume.

6.51.1 Detailed Description

Represents a rigid body with a volume (BoundingBox)

6.51.2 Constructor & Destructor Documentation

6.51.2.1 RigidBodyVolume()

```
RigidBodyVolume::RigidBodyVolume (
            NodeGameSG ∗ ng,
            bool character = false )
```

Constructor of a RigidBodyVolume with a NodeGameSG and configurable coefficients.

Parameters

ng	
mass	
friction	
cor	

6.51.2.2 ∼RigidBodyVolume()

```
RigidBodyVolume::∼RigidBodyVolume ( ) [default]
```

Destructor of a RigidBodyVolume.

6.51.3 Member Function Documentation

6.51.3.1 action()

```
void RigidBodyVolume::action (
            PhysicsSystem * ps,
            const Collision & collision,
            float delta_time )
```

Call the action function of each RigidBodyBehavior of the RigidBodyVolume.

Parameters

ps	
delta_time	

6.51.3.2 add_behavior()

```
void RigidBodyVolume::add_behavior (
            RigidBodyBehavior * behavior )
```

Add a RigidBodyBehavior to the RigidBodyVolume.

Parameters

behavior	

6.51.3.3 find_data_collision()

```
Collision RigidBodyVolume::find_data_collision (
            RigidBodyVolume & rbv )
```

Compute if there is a Collision between this RigidBodyVolume and another.

Parameters

rbv	

Returns

collision

6.51.3.4 get_movement_behavior()

```
MovementBehavior * RigidBodyVolume::get_movement_behavior ( )
```

Getter of the MovementBehavior.

Returns

movement behavior

6.51.3.5 get_node()

```
NodeGameSG * RigidBodyVolume::get_node ( )
```

Getter of the NodeGameSG.

Returns

node

6.51.3.6 has_movement_behavior()

```
bool RigidBodyVolume::has_movement_behavior ( )
```

Getter of whether or not the node has a MovementBehavior.

Returns

has movement behavior

6.51.3.7 update_physics()

```
void RigidBodyVolume::update_physics (
            float delta_time )
```

Call the update_physics function of each RigidBodyBehavior of the RigidBodyVolume.

Parameters

delta_time	

6.51.3.8 update_render()

```
void RigidBodyVolume::update_render (
            float delta_time,
            ODE * ode )
```

Call the update_render function of each RigidBodyBehavior of the RigidBodyVolume.

Parameters

delta_time	
ode	

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

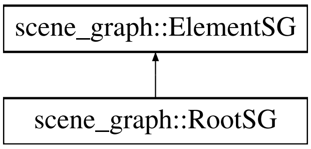
- src/physics/RigidBodyVolume.hpp
- src/physics/RigidBodyVolume.cpp

6.52 scene_graph::RootSG Class Reference

Root of the scene graph.

`#include <RootSG.hpp>`

Inheritance diagram for scene_graph::RootSG:



Public Member Functions

- glm::mat4 get_matrix_recursive (TransformDirty ∗dirty, bool inverse) override
  Give the matrix for an extern object (like a child) (recursive function)

Additional Inherited Members

6.52.1 Detailed Description

Root of the scene graph.

6.52.2 Member Function Documentation

6.52.2.1 get_matrix_recursive()

```
glm::mat4 RootSG::get_matrix_recursive (
            TransformDirty * dirty,
            bool inverse )  [override], [virtual]
```

Give the matrix for an extern object (like a child) (recursive function)

Returns

matrix

Implements scene_graph::ElementSG.

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

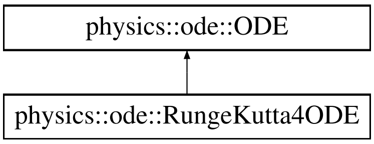
- src/scene_graph/RootSG.hpp
- src/scene_graph/RootSG.cpp

6.53 physics::ode::RungeKutta4ODE Class Reference

Runge Kutta Ordinary Differential Equation.

`#include <RungeKutta4ODE.hpp>`

Inheritance diagram for physics::ode::RungeKutta4ODE:



Public Member Functions

- RungeKutta4ODE ()
- void update (RigidBodyVolume ∗rbv, float delta_time) override

  Update position and rotation of a RigidBodyVolume with an ODE equation.

Additional Inherited Members

6.53.1 Detailed Description

Runge Kutta Ordinary Differential Equation.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 RungeKutta4ODE()

```
RungeKutta4ODE::RungeKutta4ODE ( )
```

6.53.3 Member Function Documentation

6.53.3.1 update()

```
void RungeKutta4ODE::update (
            RigidBodyVolume * rbv,
            float delta_time ) [override], [virtual]
```

Update position and rotation of a RigidBodyVolume with an ODE equation.

Parameters

rbv	
delta_time	

Implements physics::ode::ODE.

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

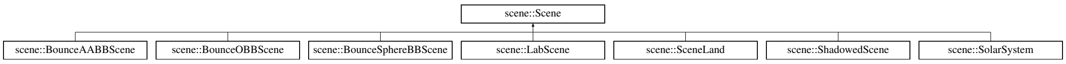
- src/physics/ode/RungeKutta4ODE.hpp
- src/physics/ode/RungeKutta4ODE.cpp

6.54 scene::Scene Class Reference

Base Scene (Abstract)

```
#include <Scene.hpp>
```

Inheritance diagram for scene::Scene:



Public Member Functions

- Scene (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_shader←_path, float mult_physics=1.0f)

  Constructor of the Scene with the paths to the shader files.
- virtual void setup ()

  Set up to load and compute datas of the Scene.
- Shaders ∗ get_shaders () const

  Getter of the shaders.
- virtual void update (float delta_time)

  Render the Scene.
- virtual void update_physics (float delta_time)

  Update the PhysicsSystem.
- virtual ∼Scene ()

  Destructor of the Scene.
- void render (bool allow_debug=true, Shaders ∗shaders=nullptr)

  Render the Scene.

Static Public Attributes

- static const int NB_MAX_LIGHTS = 10

Protected Member Functions

- virtual void process_input (float delta_time)=0

  Process the input of the user to have actions on the Scene.
- void load_lights ()

  Load the lights in the shaders.
- void load_projection_matrix ()

  Load the Projection Matrix.
- void adapt_viewport ()

  Adapt the Viewport to the window.

Protected Attributes

- float m_fovy {}
- float m_z_near {}
- float m_z_far {}
- GLFWwindow ∗ m_window
- RootSG ∗ m_root {}
- std::vector< NodeGameSG ∗ > m_cameras
- std::vector< NodeGameSG ∗ > m_lights
- int m_camera_index {}
- MainShaders ∗ m_shaders {}
- PhysicsSystem ∗ m_physics_system {}
- float m_timer = 0

6.54.1 Detailed Description

Base Scene (Abstract)

6.54.2 Constructor & Destructor Documentation

6.54.2.1 Scene()

```
Scene::Scene (
          GLFWwindow * window,
          const std::string & vertex_shader_path,
          const std::string & fragment_shader_path,
          float mult_physics = 1.0f )
```

Constructor of the Scene with the paths to the shader files.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.54.2.2 ∼Scene()

```
Scene::∼Scene ( ) [virtual]
```

Destructor of the Scene.

6.54.3 Member Function Documentation

6.54.3.1 adapt_viewport()

```
void Scene::adapt_viewport ( ) [protected]
```

Adapt the Viewport to the window.

6.54.3.2 get_shaders()

```
Shaders * Scene::get_shaders ( ) const
```

Getter of the shaders.

Returns

6.54.3.3 load_lights()

```
void Scene::load_lights ( ) [protected]
```

Load the lights in the shaders.

6.54.3.4 load_projection_matrix()

```
void Scene::load_projection_matrix ( ) [protected]
```

Load the Projection Matrix.

6.54.3.5 process_input()

```
virtual void scene::Scene::process_input (
             float delta_time ) [protected], [pure virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implemented in scene::SolarSystem, scene::LabScene, scene::BounceSphereBBScene, scene::BounceAABBScene, scene::BounceOBBScene, scene::SceneLand, and scene::ShadowedScene.

6.54.3.6 render()

```
void Scene::render (
             bool allow_debug = true,
             Shaders * shaders = nullptr )
```

Render the Scene.

Parameters

allow_debug	
other_shaders	

6.54.3.7 setup()

```
void Scene::setup ( )  [virtual]
```

Set up to load and compute datas of the Scene.

6.54.3.8 update()

```
void Scene::update (
             float delta_time )  [virtual]
```

Render the Scene.

Parameters

delta_time	

Reimplemented in scene::SolarSystem, and scene::LabScene.

6.54.3.9 update_physics()

```
void Scene::update_physics (
             float delta_time )  [virtual]
```

Update the PhysicsSystem.

Parameters

delta_time	

6.54.4 Member Data Documentation

6.54.4.1 m_camera_index

```
int scene::Scene::m_camera_index {}  [protected]
```

6.54.4.2 m_cameras

```
std::vector<NodeGameSG *> scene::Scene::m_cameras  [protected]
```

6.54.4.3 m_fovy

```
float scene::Scene::m_fovy {}  [protected]
```

6.54.4.4 m_lights

```
std::vector<NodeGameSG *> scene::Scene::m_lights  [protected]
```

6.54.4.5 m_physics_system

```
PhysicsSystem* scene::Scene::m_physics_system {}  [protected]
```

6.54.4.6 m_root

```
RootSG* scene::Scene::m_root {}  [protected]
```

6.54.4.7 m_shaders

```
MainShaders* scene::Scene::m_shaders {}  [protected]
```

6.54.4.8 m_timer

```
float scene::Scene::m_timer = 0  [protected]
```

6.54.4.9 m_window

```
GLFWwindow* scene::Scene::m_window  [protected]
```

6.54.4.10 m_z_far

```
float scene::Scene::m_z_far {}  [protected]
```

6.54.4.11 m_z_near

```
float scene::Scene::m_z_near {}  [protected]
```

6.54.4.12 NB_MAX_LIGHTS

```
const int scene::Scene::NB_MAX_LIGHTS = 10  [static]
```

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

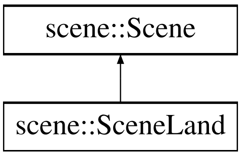
- src/scene/Scene.hpp
- src/scene/Scene.cpp

6.55 scene::SceneLand Class Reference

Scene with a land and a ball on it.

```
#include <SceneLand.hpp>
```

Inheritance diagram for scene::SceneLand:



Public Member Functions

- SceneLand (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_shader_path, float mult_physics=1.0f)

  Constructor of the SceneLand.

Static Public Attributes

- static constexpr const char ∗ HM_LAND_LOC_NAME = "hm_land"
- static constexpr const char ∗ HAS_HM_LOC_NAME = "has_hm"

Protected Member Functions

- void process_input (float delta_time) override

  Process the input of the user to have actions on the Scene.

Additional Inherited Members

6.55.1 Detailed Description

Scene with a land and a ball on it.

6.55.2 Constructor & Destructor Documentation

6.55.2.1 SceneLand()

```
SceneLand::SceneLand (
            GLFWwindow * window,
            const std::string & vertex_shader_path,
            const std::string & fragment_shader_path,
            float mult_physics = 1.0f )
```

Constructor of the SceneLand.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.55.3 Member Function Documentation

6.55.3.1 process_input()

```
void SceneLand::process_input (
            float delta_time )  [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

6.55.4 Member Data Documentation

6.55.4.1 HAS_HM_LOC_NAME

```
constexpr const char* scene::SceneLand::HAS_HM_LOC_NAME = "has_hm"  [static], [constexpr]
```

6.55.4.2 HM_LAND_LOC_NAME

```
constexpr const char* scene::SceneLand::HM_LAND_LOC_NAME = "hm_land"  [static], [constexpr]
```

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

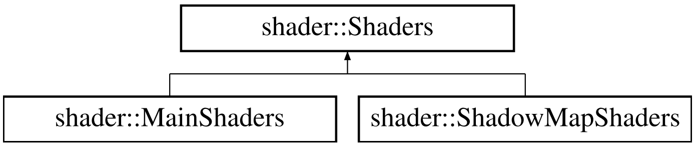
- src/scene/SceneLand.hpp
- src/scene/SceneLand.cpp

6.56 shader::Shaders Class Reference

Group the VAOManager, the ShadersDataManager and the TextureManager.

```
#include <Shaders.hpp>
```

Inheritance diagram for shader::Shaders:



Public Member Functions

- GLuint get_program_id () const

  Getter of the program id.
- void use () const

  Use the program.
- ShadersDataManager ∗ get_shader_data_manager ()

  Getter of the ShadersDataManager.
- TextureManager ∗ get_texture_manager ()

  Getter of the TextureManager.
- virtual void load_location () const =0

  Load all the locations thanks to the ShadersDataManager.

Protected Member Functions

- Shaders (const char ∗vertex_file_path, const char ∗fragment_file_path)

  Constructor of the Shaders given paths to the shaders files.

Protected Attributes

- GLuint m_program_id
- ShadersDataManager ∗ m_shader_data_manager
- TextureManager ∗ m_texture_manager

6.56.1 Detailed Description

Group the VAOManager, the ShadersDataManager and the TextureManager.

6.56.2 Constructor & Destructor Documentation

6.56.2.1 Shaders()

```
Shaders::Shaders (
            const char * vertex_file_path,
            const char * fragment_file_path )  [protected]
```

Constructor of the Shaders given paths to the shaders files.

Parameters

vertex_file_path	
fragment_file_path	

6.56.3 Member Function Documentation

6.56.3.1 get_program_id()

```
GLuint Shaders::get_program_id ( ) const
```

Getter of the program id.

Returns

program_id

6.56.3.2 get_shader_data_manager()

```
ShadersDataManager * Shaders::get_shader_data_manager ( )
```

Getter of the ShadersDataManager.

Returns

shader_data_manager

6.56.3.3 get_texture_manager()

```
TextureManager * Shaders::get_texture_manager ( )
```

Getter of the TextureManager.

Returns

texture_manager

6.56.3.4 load_location()

```
virtual void shader::Shaders::load_location ( ) const  [pure virtual]
```

Load all the locations thanks to the ShadersDataManager.

Implemented in shader::ShadowMapShaders, and shader::MainShaders.

6.56.3.5 use()

```
void Shaders::use ( ) const
```

Use the program.

6.56.4 Member Data Documentation

6.56.4.1 m_program_id

```
GLuint shader::Shaders::m_program_id  [protected]
```

6.56.4.2 m_shader_data_manager

```
ShadersDataManager* shader::Shaders::m_shader_data_manager  [protected]
```

6.56.4.3 m_texture_manager

```
TextureManager* shader::Shaders::m_texture_manager  [protected]
```

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

- src/shader/Shaders.hpp
- src/shader/Shaders.cpp

6.57 shader::ShadersDataManager Class Reference

Manager of the locations in shaders.

```
#include <ShadersDataManager.hpp>
```

Public Member Functions

- void load_matrices_locations (GLuint program_id)

  Load the location of the matrices.
- void load_view_pos_location (GLuint program_id)

  Load the location of the view position.
- void load_node_on_top_locations (GLuint program_id)

  Load the location of the node on top datas.
- void load_lights_const (GLuint program_id)

  Load the location and the datas of the Light Type Constants.
- void load_lights_locations (GLuint program_id)

  Load the locations of the Light system.
- void load_debug_const (GLuint program_id)

  Load the locations of the debug rendering constant.
- void load_material_const (GLuint program_id)

  Load the location and the datas of the Material Type Constants.
- void load_material_locations (GLuint program_id)

  Load the locations of the Material system.
- void load_custom_uniform_location (GLuint program_id, const std::string &name)

  Load the location for a custom data.
- GLint get_location (const std::string &name)

  Getter of a data given the name of it.
- void load_lights (GLuint program_id, LightShader lights_shader[ ], int size_lights)

  Load the data of the lights given a list of LightShader.
- void load_shadow_map_matrix_location (GLuint program_id)

  Load the location of the matrix of the shadow map.
- void load_shadow_maps_location (GLuint program_id)

  Load the locations of the shadow map.

Static Public Attributes

- static constexpr const char ∗ MODEL_MAT_LOC_NAME = "model_mat"
- static constexpr const char ∗ VIEW_MAT_LOC_NAME = "view_mat"
- static constexpr const char ∗ PROJ_MAT_LOC_NAME = "projection_mat"
- static constexpr const char ∗ NORMAL_MODEL_MAT_LOC_NAME = "normal_model_mat"
- static constexpr const char ∗ VIEW_POS_LOC_NAME = "view_pos"
- static constexpr const char ∗ NB_LIGTHS_LOC_NAME = "nb_lights"
- static constexpr const char ∗ BLOCK_INDEX_LIGHTS_LOC_NAME = "lights_buffer"
- static constexpr const char ∗ MATERIAL_TYPE_COLOR_LOC_NAME = "MATERIAL_TYPE_COLOR"
- static constexpr const char ∗ MATERIAL_TYPE_TEXTURE_LOC_NAME = "MATERIAL_TYPE_TEXTURE"
- static constexpr const char ∗ MATERIAL_TYPE_LOC_NAME = "material.type"
- static constexpr const char ∗ MATERIAL_SHININESS_LOC_NAME = "material.shininess"
- static constexpr const char ∗ MATERIAL_AMBIENT_LOC_NAME = "material.ambient"
- static constexpr const char ∗ MATERIAL_DIFFUSE_LOC_NAME = "material.diffuse"
- static constexpr const char ∗ MATERIAL_SPECULAR_LOC_NAME = "material.specular"
- static constexpr const char ∗ MATERIAL_DIFFUSE_TEXTURE_LOC_NAME = "material.diffuse_texture"
- static constexpr const char ∗ MATERIAL_SPECULAR_TEXTURE_LOC_NAME = "material.specular_texture"
- static constexpr const char ∗ LIGHT_TYPE_DIRECTIONAL_LOC_NAME = "LIGHT_TYPE_DIRECTIONAL"
- static constexpr const char ∗ LIGHT_TYPE_POINT_LOC_NAME = "LIGHT_TYPE_POINT"
- static constexpr const char ∗ LIGHT_TYPE_SPOT_LOC_NAME = "LIGHT_TYPE_SPOT"
- static constexpr const char ∗ IS_NODE_ON_TOP_LOC_NAME = "is_node_on_top"
- static constexpr const char ∗ ON_TOP_POSITION_LOC_NAME = "on_top_position"
- static constexpr const char ∗ ON_TOP_NORMAL_LOC_NAME = "on_top_normal"
- static constexpr const char ∗ ON_TOP_UV_LOC_NAME = "on_top_uv"
- static constexpr const char ∗ ON_TOP_MODEL_LOC_NAME = "on_top_model"
- static constexpr const char ∗ ON_TOP_HEIGHT_ADJUSTMENT_LOC_NAME = "on_top_height_↩adjustement"
- static constexpr const char ∗ SHADOW_MAP_ARRAY_LOC_NAME = "shadow_maps"
- static constexpr const char ∗ SHADOW_MAP_DEPTH_VP_MAT_LOC_NAME = "depth_vp_mat"
- static constexpr const char ∗ DEBUG_RENDERING_LOC_NAME = "debug_rendering"
- static constexpr const char ∗ DEBUG_RENDERING_COLOR_LOC_NAME = "debug_rendering_color"

6.57.1 Detailed Description

Manager of the locations in shaders.

6.57.2 Member Function Documentation

6.57.2.1 get_location()

```
GLint ShadersDataManager::get_location (
            const std::string & name )
```

Getter of a data given the name of it.

Parameters

name	

Returns

location

6.57.2.2 load_custom_uniform_location()

```
void ShadersDataManager::load_custom_uniform_location (
            GLuint program_id,
            const std::string & name )
```

Load the location for a custom data.

Parameters

program↩ _id	
name	

6.57.2.3 load_debug_const()

```
void ShadersDataManager::load_debug_const (
            GLuint program_id )
```

Load the locations of the debug rendering constant.

Parameters

program↩ _id	

6.57.2.4 load_lights()

```
void ShadersDataManager::load_lights (
            GLuint program_id,
            LightShader lights_shader[ ],
            int size_lights )
```

Load the data of the lights given a list of LightShader.

Parameters

program_id	
lights_shader	
size_lights	

6.57.2.5 load_lights_const()

```
void ShadersDataManager::load_lights_const (
            GLuint program_id )
```

Load the location and the datas of the Light Type Constants.

Parameters

program↩ _id	

6.57.2.6 load_lights_locations()

```
void ShadersDataManager::load_lights_locations (
            GLuint program_id )
```

Load the locations of the Light system.

Parameters

program↩ _id	

6.57.2.7 load_material_const()

```
void ShadersDataManager::load_material_const (
            GLuint program_id )
```

Load the location and the datas of the Material Type Constants.

Parameters

program↩ _id	

6.57.2.8 load_material_locations()

```
void ShadersDataManager::load_material_locations (
            GLuint program_id )
```

Load the locations of the Material system.

Parameters

program↩ _id	

6.57.2.9 load_matrices_locations()

```
void ShadersDataManager::load_matrices_locations (
            GLuint program_id )
```

Load the location of the matrices.

Parameters

program↩ _id	

6.57.2.10 load_node_on_top_locations()

```
void ShadersDataManager::load_node_on_top_locations (
            GLuint program_id )
```

Load the location of the node on top datas.

Parameters

program↩ _id	

6.57.2.11 load_shadow_map_matrix_location()

```
void ShadersDataManager::load_shadow_map_matrix_location (
            GLuint program_id )
```

Load the location of the matrix of the shadow map.

Parameters

program↩ _id	

6.57.2.12 load_shadow_maps_location()

```
void ShadersDataManager::load_shadow_maps_location (
            GLuint program_id )
```

Load the locations of the shadow map.

Parameters

program↩ _id	

6.57.2.13 load_view_pos_location()

```
void ShadersDataManager::load_view_pos_location (
            GLuint program_id )
```

Load the location of the view position.

Parameters

program↩ _id	

6.57.3 Member Data Documentation

6.57.3.1 BLOCK_INDEX_LIGHTS_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::BLOCK_INDEX_LIGHTS_LOC_NAME = "lights_↩
buffer" [static], [constexpr]
```

6.57.3.2 DEBUG_RENDERING_COLOR_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::DEBUG_RENDERING_COLOR_LOC_NAME = "debug_↩
rendering_color" [static], [constexpr]
```

6.57.3.3 DEBUG_RENDERING_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::DEBUG_RENDERING_LOC_NAME = "debug_rendering"
[static], [constexpr]
```

6.57.3.4 IS_NODE_ON_TOP_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::IS_NODE_ON_TOP_LOC_NAME = "is_node_on_top"
[static], [constexpr]
```

6.57.3.5 LIGHT_TYPE_DIRECTIONAL_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::LIGHT_TYPE_DIRECTIONAL_LOC_NAME = "LIGHT_TY←
PE_DIRECTIONAL"  [static], [constexpr]
```

6.57.3.6 LIGHT_TYPE_POINT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::LIGHT_TYPE_POINT_LOC_NAME = "LIGHT_TYPE_POI←
NT"  [static], [constexpr]
```

6.57.3.7 LIGHT_TYPE_SPOT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::LIGHT_TYPE_SPOT_LOC_NAME = "LIGHT_TYPE_SPOT"
[static], [constexpr]
```

6.57.3.8 MATERIAL_AMBIENT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_AMBIENT_LOC_NAME = "material.←
ambient"  [static], [constexpr]
```

6.57.3.9 MATERIAL_DIFFUSE_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_DIFFUSE_LOC_NAME = "material.←
diffuse"  [static], [constexpr]
```

6.57.3.10 MATERIAL_DIFFUSE_TEXTURE_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_DIFFUSE_TEXTURE_LOC_NAME = "material.↩
diffuse_texture" [static], [constexpr]
```

6.57.3.11 MATERIAL_SHININESS_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_SHININESS_LOC_NAME = "material.↩
shininess" [static], [constexpr]
```

6.57.3.12 MATERIAL_SPECULAR_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_SPECULAR_LOC_NAME = "material.↩
specular" [static], [constexpr]
```

6.57.3.13 MATERIAL_SPECULAR_TEXTURE_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_SPECULAR_TEXTURE_LOC_NAME = "material.↩
specular_texture" [static], [constexpr]
```

6.57.3.14 MATERIAL_TYPE_COLOR_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_TYPE_COLOR_LOC_NAME = "MATERIAL_TY↩
PE_COLOR" [static], [constexpr]
```

6.57.3.15 MATERIAL_TYPE_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_TYPE_LOC_NAME = "material.type"
[static], [constexpr]
```

6.57.3.16 MATERIAL_TYPE_TEXTURE_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MATERIAL_TYPE_TEXTURE_LOC_NAME = "MATERIAL_↩
TYPE_TEXTURE" [static], [constexpr]
```

6.57.3.17 MODEL_MAT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::MODEL_MAT_LOC_NAME = "model_mat"  [static],
[constexpr]
```

6.57.3.18 NB_LIGTHS_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::NB_LIGTHS_LOC_NAME = "nb_lights"  [static],
[constexpr]
```

6.57.3.19 NORMAL_MODEL_MAT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::NORMAL_MODEL_MAT_LOC_NAME = "normal_model_↩
mat"  [static], [constexpr]
```

6.57.3.20 ON_TOP_HEIGHT_ADJUSTMENT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::ON_TOP_HEIGHT_ADJUSTMENT_LOC_NAME = "on_top↩
_height_adjustement"  [static], [constexpr]
```

6.57.3.21 ON_TOP_MODEL_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::ON_TOP_MODEL_LOC_NAME = "on_top_model"  [static],
[constexpr]
```

6.57.3.22 ON_TOP_NORMAL_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::ON_TOP_NORMAL_LOC_NAME = "on_top_normal"
[static], [constexpr]
```

6.57.3.23 ON_TOP_POSITION_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::ON_TOP_POSITION_LOC_NAME = "on_top_position"
[static], [constexpr]
```

6.57.3.24 ON_TOP_UV_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::ON_TOP_UV_LOC_NAME = "on_top_uv"  [static],
[constexpr]
```

6.57.3.25 PROJ_MAT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::PROJ_MAT_LOC_NAME = "projection_mat"  [static],
[constexpr]
```

6.57.3.26 SHADOW_MAP_ARRAY_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::SHADOW_MAP_ARRAY_LOC_NAME = "shadow_maps"
[static], [constexpr]
```

6.57.3.27 SHADOW_MAP_DEPTH_VP_MAT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::SHADOW_MAP_DEPTH_VP_MAT_LOC_NAME = "depth_↩
vp_mat"  [static], [constexpr]
```

6.57.3.28 VIEW_MAT_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::VIEW_MAT_LOC_NAME = "view_mat"  [static],
[constexpr]
```

6.57.3.29 VIEW_POS_LOC_NAME

```
constexpr const char* shader::ShadersDataManager::VIEW_POS_LOC_NAME = "view_pos"  [static],
[constexpr]
```

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

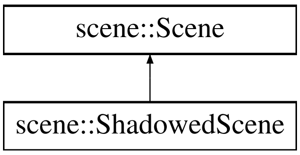
- src/shader/ShadersDataManager.hpp
- src/shader/ShadersDataManager.cpp

6.58 scene::ShadowedScene Class Reference

Scene with Collisions.

```
#include <ShadowedScene.hpp>
```

Inheritance diagram for scene::ShadowedScene:



Public Member Functions

- ShadowedScene (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_shader_path, float mult_physics=1.0f)

  Constructor of the ShadowedScene.

Protected Member Functions

- void process_input (float delta_time) override

  Process the input of the user to have actions on the Scene.

Protected Attributes

- NodeGameSG ∗ m_sphere_light

Additional Inherited Members

6.58.1 Detailed Description

Scene with Collisions.

6.58.2 Constructor & Destructor Documentation

6.58.2.1 ShadowedScene()

```
ShadowedScene::ShadowedScene (
            GLFWwindow ∗ window,
            const std::string & vertex_shader_path,
            const std::string & fragment_shader_path,
            float mult_physics = 1.0f )
```

Constructor of the ShadowedScene.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.58.3 Member Function Documentation

6.58.3.1 process_input()

```
void ShadowedScene::process_input (
            float delta_time )  [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

6.58.4 Member Data Documentation

6.58.4.1 m_sphere_light

```
NodeGameSG* scene::ShadowedScene::m_sphere_light  [protected]
```

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

- src/scene/ShadowedScene.hpp
- src/scene/ShadowedScene.cpp

6.59 shader::ShadowMap Class Reference

Represents a Shadow Map.

```
#include <ShadowMap.hpp>
```

Public Member Functions

- ShadowMap (int w, int h, GLuint id_texture_unit_depth_map)

  Constructor of a ShadowMap with size and id_texture.
- ∼ShadowMap ()

  Destructor of a ShadowMap.
- void bind () const

  Bind the ShadowMap for writing.
- void activate_texture () const

  Activate the texture.
- GLuint get_depth_map_id_texture () const

  Getter of the id texture of the depth map.
- int get_width () const

  Getter of the width.
- int get_height () const

  Getter of the height.
- void print_in_img (const char ∗path_name) const

  Print the ShadowMap in an img.

Static Public Member Functions

- static void unbind_bound_shadow_map ()

  Unbind the bound ShadowMap.

6.59.1 Detailed Description

Represents a Shadow Map.

6.59.2 Constructor & Destructor Documentation

6.59.2.1 ShadowMap()

```
ShadowMap::ShadowMap (
          int w,
          int h,
          GLuint id_texture_unit_depth_map )
```

Constructor of a ShadowMap with size and id_texture.

Parameters

w	
h	
id_texture_unit_depth_map	

6.59.2.2 ∼ShadowMap()

```
ShadowMap::∼ShadowMap ( )
```

Destructor of a ShadowMap.

6.59.3 Member Function Documentation

6.59.3.1 activate_texture()

```
void ShadowMap::activate_texture ( ) const
```

Activate the texture.

6.59.3.2 bind()

```
void ShadowMap::bind ( ) const
```

Bind the ShadowMap for writing.

6.59.3.3 get_depth_map_id_texture()

```
GLuint ShadowMap::get_depth_map_id_texture ( ) const
```

Getter of the id texture of the depth map.

Returns

id texture

6.59.3.4 get_height()

```
int ShadowMap::get_height ( ) const
```

Getter of the height.

Returns

height

6.59.3.5 get_width()

```
int ShadowMap::get_width ( ) const
```

Getter of the width.

Returns

width

6.59.3.6 print_in_img()

```
void ShadowMap::print_in_img (
            const char * path_name ) const
```

Print the ShadowMap in an img.

Parameters

name	

6.59.3.7 unbind_bound_shadow_map()

```
void ShadowMap::unbind_bound_shadow_map ( )  [static]
```

Unbind the bound ShadowMap.

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

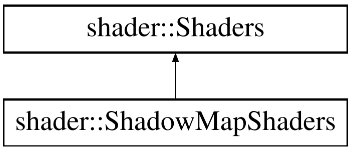
- src/shader/ShadowMap.hpp
- src/shader/ShadowMap.cpp

6.60 shader::ShadowMapShaders Class Reference

Manager of the Shadow Mapping Shaders.

```
#include <ShadowMapShaders.hpp>
```

Inheritance diagram for shader::ShadowMapShaders:



Public Member Functions

- ShadowMapShaders (const char ∗vertex_file_path, const char ∗fragment_file_path)

  Constructor of the ShadowMapShaders.
- void load_location () const override

  Load all the locations thanks to the ShadersDataManager.

Additional Inherited Members

6.60.1 Detailed Description

Manager of the Shadow Mapping Shaders.

6.60.2 Constructor & Destructor Documentation

6.60.2.1 ShadowMapShaders()

```
ShadowMapShaders::ShadowMapShaders (
            const char * vertex_file_path,
            const char * fragment_file_path )
```

Constructor of the ShadowMapShaders.

Parameters

vertex_file_path	
fragment_file_path	

6.60.3 Member Function Documentation

6.60.3.1 load_location()

```
void ShadowMapShaders::load_location ( ) const  [override], [virtual]
```

Load all the locations thanks to the ShadersDataManager.

Implements shader::Shaders.

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

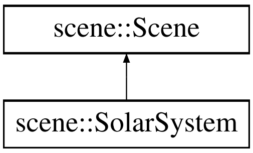
- src/shader/ShadowMapShaders.hpp
- src/shader/ShadowMapShaders.cpp

6.61 scene::SolarSystem Class Reference

Scene with the sun, the earth, the moon end the sky.

```
#include <SolarSystem.hpp>
```

Inheritance diagram for scene::SolarSystem:



Public Member Functions

- SolarSystem (GLFWwindow ∗window, const std::string &vertex_shader_path, const std::string &fragment_shader_path, float mult_physics=1.0f)

    Constructor of the SolarSystem.
- void update (float delta_time) override

    Render the Scene.

Protected Member Functions

- void process_input (float delta_time) override

    Process the input of the user to have actions on the Scene.
- void load_type_star_location () const

    Load the star type to the shader.

Additional Inherited Members

6.61.1 Detailed Description

Scene with the sun, the earth, the moon end the sky.

6.61.2 Constructor & Destructor Documentation

6.61.2.1 SolarSystem()

```
SolarSystem::SolarSystem (
            GLFWwindow * window,
            const std::string & vertex_shader_path,
            const std::string & fragment_shader_path,
            float mult_physics = 1.0f )
```

Constructor of the SolarSystem.

Parameters

window	
vertex_shader_path	
fragment_shader_path	
mult_physics	

6.61.3 Member Function Documentation

6.61.3.1 load_type_star_location()

```
void SolarSystem::load_type_star_location ( ) const  [protected]
```

Load the star type to the shader.

6.61.3.2 process_input()

```
void SolarSystem::process_input (
             float delta_time )  [override], [protected], [virtual]
```

Process the input of the user to have actions on the Scene.

Parameters

delta_time	

Implements scene::Scene.

6.61.3.3 update()

```
void SolarSystem::update (
             float delta_time )  [override], [virtual]
```

Render the Scene.

Parameters

delta_time	

Reimplemented from scene::Scene.

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

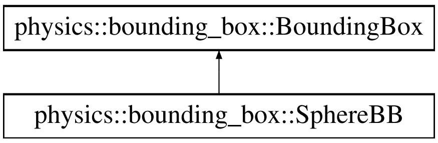
- src/scene/SolarSystem.hpp
- src/scene/SolarSystem.cpp

6.62 physics::bounding_box::SphereBB Class Reference

Sphere BoundingBox.

```
#include <SphereBB.hpp>
```

Inheritance diagram for physics::bounding_box::SphereBB:


physics::bounding_box::BoundingBox
physics::bounding_box::SphereBB

Public Member Functions

- SphereBB ()

  Constructor of SphereBB.
- void compute (std::vector< glm::vec3 > vertices) override

  Compute the BoundingBox according to vertices.
- float get_radius () const

  Getter of the radius of the SphereBB.
- void apply_transform (glm::mat4 matrix) override

  Apply the Matrix to the BoundingBox.
- Collision get_data_collision (SphereBB ∗bb) override

  Compute the Collision with a SphereBB.
- Collision get_data_collision (RCBB ∗bb) override

  Compute the Collision with an RCBB.
- AABB ∗ to_AABB () const override

  Convert the BoundingBox to an AABB.
- Interval get_interval (glm::vec3 axis) override

  Get the Interval clip on an axis.
- std::vector< glm::vec3 > to_vertices () const override

  Give the vertices that compose the BoundingBox.
- float is_intersected (Ray) override

  Returns intersection distance along ray direction between ray and boundingbox.
- glm::vec3 closest_point (glm::vec3 pt) const override

  Returns return closest point on bounding box to pt.
- glm::vec3 get_tensor () override

  Getter of the tensor matrix.
- float get_max_dist () override

  Getter of the max distance between two points in a bb.

Additional Inherited Members

6.62.1 Detailed Description

Sphere BoundingBox.

6.62.2 Constructor & Destructor Documentation

6.62.2.1 SphereBB()

```
SphereBB::SphereBB ( )
```

Constructor of SphereBB.

6.62.3 Member Function Documentation

6.62.3.1 apply_transform()

```
void SphereBB::apply_transform (
             glm::mat4 matrix )  [override], [virtual]
```

Apply the Matrix to the BoundingBox.

Parameters

matrix	

Implements physics::bounding_box::BoundingBox.

6.62.3.2 closest_point()

```
glm::vec3 SphereBB::closest_point (
             glm::vec3 pt ) const  [override], [virtual]
```

Returns return closest point on bounding box to pt.

Returns

Implements physics::bounding_box::BoundingBox.

6.62.3.3 compute()

```
void SphereBB::compute (
            std::vector< glm::vec3 > vertices )  [override], [virtual]
```

Compute the BoundingBox according to vertices.

Parameters

vertices	

Implements physics::bounding_box::BoundingBox.

6.62.3.4 get_data_collision() [1/2]

```
Collision SphereBB::get_data_collision (
            RCBB * bb )  [override], [virtual]
```

Compute the Collision with an RCBB.

Parameters

bb	

Returns

collision

Implements physics::bounding_box::BoundingBox.

6.62.3.5 get_data_collision() [2/2]

```
Collision SphereBB::get_data_collision (
            SphereBB * bb )  [override], [virtual]
```

Compute the Collision with a SphereBB.

Parameters

bb	

Returns

collision

Implements physics::bounding_box::BoundingBox.

6.62.3.6 get_interval()

```
Interval SphereBB::get_interval (
            glm::vec3 axis )  [override], [virtual]
```

Get the Interval clip on an axis.

Parameters

axis	

Returns

interval

Implements physics::bounding_box::BoundingBox.

6.62.3.7 get_max_dist()

```
float SphereBB::get_max_dist ( )  [override], [virtual]
```

Getter of the max distance between two points in a bb.

Returns

float distance

Implements physics::bounding_box::BoundingBox.

6.62.3.8 get_radius()

```
float SphereBB::get_radius ( ) const
```

Getter of the radius of the SphereBB.

Returns

6.62.3.9 get_tensor()

```
glm::vec3 SphereBB::get_tensor ( )  [override], [virtual]
```

Getter of the tensor matrix.

Returns

tensor matrix

Implements physics::bounding_box::BoundingBox.

6.62.3.10 is_intersected()

```
float SphereBB::is_intersected (
            Ray  )  [override], [virtual]
```

Returns intersection distance along ray direction between ray and boundingbox.

Returns

Implements physics::bounding_box::BoundingBox.

6.62.3.11 to_AABB()

```
AABB * SphereBB::to_AABB ( ) const  [override], [virtual]
```

Convert the BoundingBox to an AABB.

Returns

aabb

Implements physics::bounding_box::BoundingBox.

6.62.3.12 to_vertices()

```
std::vector< glm::vec3 > SphereBB::to_vertices ( ) const  [override], [virtual]
```

Give the vertices that compose the BoundingBox.

Returns

Implements physics::bounding_box::BoundingBox.

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

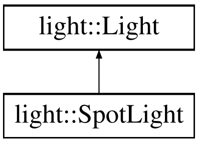
- src/physics/bounding_box/SphereBB.hpp
- src/physics/bounding_box/SphereBB.cpp

6.63 light::SpotLight Class Reference

Light with a direction, a position and angles of diffusion.

```
#include <SpotLight.hpp>
```

Inheritance diagram for light::SpotLight:



Public Member Functions

- SpotLight (glm::vec3 ambient, glm::vec3 diffuse, glm::vec3 specular, GLuint id_texture_shadow_map, float inner_cut_off_angle=20.f, float outer_cut_off_angle=25.f, int resol=1000, float z_near=1.0f, float z_far=1000.f, float bias=0.001f, float constant_attenuation=0.f, float linear_attenuation=0.f, float quadratic_attenuation=0.f)

  Constructor of a Spot Light (has a DirectionLightBehavior, a PositionLightBehavior and a SpotLightBehavior)
- void to_light_info (LightInfo ∗light_struct, glm::mat4 model_mat) override

  fill the data into a LightInfo

Additional Inherited Members

6.63.1 Detailed Description

Light with a direction, a position and angles of diffusion.

6.63.2 Constructor & Destructor Documentation

6.63.2.1 SpotLight()

```
SpotLight::SpotLight (
            glm::vec3 ambient,
            glm::vec3 diffuse,
            glm::vec3 specular,
            GLuint id_texture_shadow_map,
            float inner_cut_off_angle = 20.f,
            float outer_cut_off_angle = 25.f,
            int resol = 1000,
            float z_near = 1.0f,
            float z_far = 1000.f,
            float bias = 0.001f,
            float constant_attenuation = 0.f,
            float linear_attenuation = 0.f,
            float quadratic_attenuation = 0.f )  [explicit]
```

Constructor of a Spot Light (has a DirectionLightBehavior, a PositionLightBehavior and a SpotLightBehavior)

Parameters

ambient	
diffuse	
specular	
id_texture_shadow_map	
inner_cut_off_angle	
outer_cut_off_angle	
resol	
z_near	
z_far	
bias	
constant_attenuation	
linear_attenuation	
quadratic_attenuation	

6.63.3 Member Function Documentation

6.63.3.1 to_light_info()

```
void SpotLight::to_light_info (
            LightInfo * light_shader,
            glm::mat4 model_mat )  [override], [virtual]
```

fill the data into a LightInfo

Parameters

light_shader	
model_mat	

Reimplemented from light::Light.

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

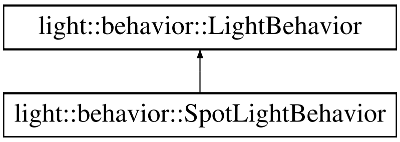
- src/light/SpotLight.hpp
- src/light/SpotLight.cpp

6.64 light::behavior::SpotLightBehavior Class Reference

Behavior of a SpotLight.

```
#include <SpotLightBehavior.hpp>
```

Inheritance diagram for light::behavior::SpotLightBehavior:


light::behavior::LightBehavior
light::behavior::SpotLightBehavior

Public Member Functions

- SpotLightBehavior (GLuint id_texture_shadow_map, float icoa=20.f, float ocoa=25.f, int resol=1000, float z↩_near=1.0f, float z_far=1000.f, float bias=0.001f)

    Constructor of a SpotLightBehavior with the inner, the outer cut off and the id texture for the shadow map.
- void apply_to (LightInfo ∗light_shader, glm::mat4 model_mat) override

    Apply the behavior on the LightShader to resolve its data's.

6.64.1 Detailed Description

Behavior of a SpotLight.

6.64.2 Constructor & Destructor Documentation

6.64.2.1 SpotLightBehavior()

```
SpotLightBehavior::SpotLightBehavior (
            GLuint id_texture_shadow_map,
            float icoa = 20.f,
            float ocoa = 25.f,
            int resol = 1000,
            float z_near = 1.0f,
            float z_far = 1000.f,
            float bias = 0.001f ) [explicit]
```

Constructor of a SpotLightBehavior with the inner, the outer cut off and the id texture for the shadow map.

Parameters

id_texture_shadow_map	
icoa	
ocoa	
resol	
z_near	
z_far	
bias	

6.64.3 Member Function Documentation

6.64.3.1 apply_to()

```
void SpotLightBehavior::apply_to (
            LightInfo * light_info,
            glm::mat4 model_mat )  [override], [virtual]
```

Apply the behavior on the LightShader to resolve its data's.

Parameters

light_info	
model_mat	

Implements light::behavior::LightBehavior.

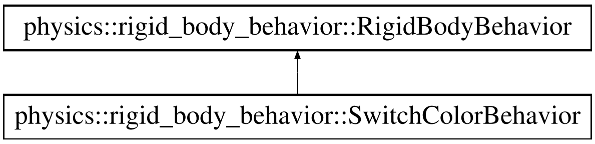
The documentation for this class was generated from the following files:

- src/light/light_behavior/SpotLightBehavior.hpp
- src/light/light_behavior/SpotLightBehavior.cpp

6.65 physics::rigid_body_behavior::SwitchColorBehavior Class Reference

```
#include <SwitchColorBehavior.hpp>
```

Inheritance diagram for physics::rigid_body_behavior::SwitchColorBehavior:


physics::rigid_body_behavior::RigidBodyBehavior
physics::rigid_body_behavior::SwitchColorBehavior

Public Member Functions

- SwitchColorBehavior (Material ∗material)

  Constructor of SwitchColorBehavior.
- void action (PhysicsSystem ∗ps, Collision collision, float delta_time) override

  Function called each time that the RigidBodyVolume collide.
- void update_physics (float delta_time) override

  Function called each time the physic is updated.
- void update_render (float delta_time, ODE ∗ode) override

  Function called each time the rendering is updated.
- void can_collide_with (RigidBodyVolume ∗rbv)

  Specify a collider that trigger action of the SwitchColorBehavior.

Additional Inherited Members

6.65.1 Constructor & Destructor Documentation

6.65.1.1 SwitchColorBehavior()

```
SwitchColorBehavior::SwitchColorBehavior (
            Material * material )  [explicit]
```

Constructor of SwitchColorBehavior.

Parameters

material	

6.65.2 Member Function Documentation

6.65.2.1 action()

```
void SwitchColorBehavior::action (
            PhysicsSystem * ps,
            Collision collision,
            float delta_time )  [override], [virtual]
```

Function called each time that the RigidBodyVolume collide.

Parameters

ps	
collision	
delta_time	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

6.65.2.2 can_collide_with()

```
void SwitchColorBehavior::can_collide_with (
            RigidBodyVolume * rbv )
```

Specify a collider that trigger action of the SwitchColorBehavior.

Parameters

rbv	

6.65.2.3 update_physics()

```
void SwitchColorBehavior::update_physics (
            float delta_time )  [override], [virtual]
```

Function called each time the physic is updated.

Parameters

delta_time	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

6.65.2.4 update_render()

```
void SwitchColorBehavior::update_render (
            float delta_time,
            ODE * ode )  [override], [virtual]
```

Function called each time the rendering is updated.

Parameters

delta_time	
ode	

Implements physics::rigid_body_behavior::RigidBodyBehavior.

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

- src/physics/rigid_body_behavior/SwitchColorBehavior.hpp
- src/physics/rigid_body_behavior/SwitchColorBehavior.cpp

6.66 shader::TextureManager Class Reference

Manager of all the texture in the shaders.

```
#include <TextureManager.hpp>
```

Public Member Functions

- TextureManager (ShadersDataManager ∗shader_data_manager)

  Constructor of the TextureManager.
- GLuint load_uniform_texture (GLuint program_id, const std::string &name, const std::string &path)

  Load a new texture in the shader.
- GLint load_texture (const std::string &path)

  Load a texture at an existant location in the shader.
- GLint get_and_increment_id_texture ()

  Getter of the next id_texture and increment it.

6.66.1 Detailed Description

Manager of all the texture in the shaders.

6.66.2 Constructor & Destructor Documentation

6.66.2.1 TextureManager()

```
TextureManager::TextureManager (
            shader::ShadersDataManager ∗ shader_data_manager ) [explicit]
```

Constructor of the TextureManager.

Parameters

shader_data_manager	

6.66.3 Member Function Documentation

6.66.3.1 get_and_increment_id_texture()

```
GLint TextureManager::get_and_increment_id_texture ( )
```

Getter of the next id_texture and increment it.

Returns

id_texture

6.66.3.2 load_texture()

```
GLint TextureManager::load_texture (
            const std::string & path )
```

Load a texture at an existant location in the shader.

Parameters

path	

Returns

id_texture

6.66.3.3 load_uniform_texture()

```
GLuint TextureManager::load_uniform_texture (
            GLuint program_id,
            const std::string & name,
            const std::string & path )
```

Load a new texture in the shader.

Parameters

program↩ _id	
name	
path	

Returns

id_texture

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

- src/shader/TextureManager.hpp
- src/shader/TextureManager.cpp

6.67 Transform Class Reference

Class representing a 4 by 4 Matrix (Translation + Rotation + Scale)

```
#include <Transform.hpp>
```

Public Member Functions

- Transform (glm::vec3 translation={0.0f, 0.0f, 0.0f}, glm::vec3 rotation={0.0f, 0.0f, 0.0f}, glm::vec3 scale={1.0f, 1.0f, 1.0f}, int order_rotation=ORDER_YXZ)

  Constructor of the Transform.
- void init (glm::vec3 translation={0.0f, 0.0f, 0.0f}, glm::vec3 rotation={0.0f, 0.0f, 0.0f}, glm::vec3 scale={1.0f, 1.0f, 1.0f}, int order_rotation=ORDER_YXZ)

  Init method also used to reset te Transform.
- void compute ()

  Compute the matrix of the Transform.
- glm::mat4 compute_lerp_with_transform (Transform transf, float k)

  Compute linear interpolation given an other Transform.
- void set_order_rotation (int order_rotation)

  Setter of the order of rotation.
- void set_translation (const glm::vec3 &new_translation)

  Setter of the translation.
- void set_rotation (const glm::vec3 &new_rotation)

  Setter of the rotation.
- void set_scale (const glm::vec3 &new_scale)

  Setter of the scale.
- void set_uniform_scale (float scale)

  Setter of the scale with an uniform scale.
- void set_matrix (const glm::mat4 &new_matrix)

  Setter of the matrix.
- const glm::vec3 & get_translation ()

  Getter of the translation.
- const glm::vec3 & get_scale ()

  Getter of the scale.
- const glm::vec3 & get_rotation ()

  Getter of the rotation.
- const glm::mat4 & get_matrix ()

  Getter of the matrix.
- glm::mat4 get_inverse ()

  Get the matrix that correspond at the inverse of the transform.
- TransformDirty ∗ is_dirty () const

  Getter of the states dirty of the Transform.
- bool is_up_to_date () const

  Getter of whether the Transform is up to date or no.
- glm::vec3 apply_to_point (glm::vec3 &v)

  Apply the Transform to a point.

- glm::vec3 apply_to_vector (glm::vec3 &v)

  Apply the Transform to a vector.
- glm::vec3 apply_to_versor (glm::vec3 &v)

  Apply the Transform to a versor.
- void apply_to_vector_of_point (std::vector< glm::vec3 > *points)

  Apply the Transform to a list of point.
- void apply_to_vector_of_vector (std::vector< glm::vec3 > *vectors)

  Apply the Transform to a list of vector.
- void apply_to_vector_of_versor (std::vector< glm::vec3 > *versors)

  Apply the Transform to a list of versor.

Static Public Member Functions

- static void matrix_to_trs (glm::mat4 matrix_to_decompose, glm::mat4 &t, glm::mat4 &r, glm::mat4 &s)

  Decompose a matrix to 3 matrices(TRS)

Protected Member Functions

- glm::mat4 local_get_matrix ()

  Compute the matrix of the Transform without saving it.
- glm::vec3 apply_to_vec3 (glm::vec3 &v, bool with_translation, bool with_normalization)

  Apply the Transform to a vec3 (General method)
- void apply_to_vector_of_vec3 (std::vector< glm::vec3 > *vects, bool with_translation, bool with_↩ normalization)

  Apply the Transform to a list of vec3 (General method)

Static Protected Member Functions

- static glm::mat4 local_get_matrix_with_values (glm::vec3 tr, glm::vec3 rot, glm::vec3 sc, int order_rotation, bool inverse=false)

  Compute a matrix with given values.

Protected Attributes

- glm::vec3 m_translate
- glm::vec3 m_rot
- glm::vec3 m_scale
- glm::mat4 m_matrix
- bool m_up_to_date {}
- TransformDirty * m_dirty {}
- int m_order_rotation {}

Friends

- bool operator== (const Transform &trsf1, const Transform &trsf2)

  Operator equal.
- bool operator!= (const Transform &trsf1, const Transform &trsf2)

  Operator inequal.

6.67.1 Detailed Description

Class representing a 4 by 4 Matrix (Translation + Rotation + Scale)

6.67.2 Constructor & Destructor Documentation

6.67.2.1 Transform()

```
Transform::Transform (
            glm::vec3 translation = {0.0f, 0.0f, 0.0f},
            glm::vec3 rotation = {0.0f, 0.0f, 0.0f},
            glm::vec3 scale = {1.0f, 1.0f, 1.0f},
            int order_rotation = ORDER_YXZ )  [explicit]
```

Constructor of the Transform.

Parameters

translation	
rotation	
scale	
order_rotation	

6.67.3 Member Function Documentation

6.67.3.1 apply_to_point()

```
glm::vec3 Transform::apply_to_point (
            glm::vec3 & v )
```

Apply the Transform to a point.

Parameters

v	

Returns

point

6.67.3.2 apply_to_vec3()

```
glm::vec3 Transform::apply_to_vec3 (
             glm::vec3 & v,
             bool with_translation,
             bool with_normalization )  [protected]
```

Apply the Transform to a vec3 (General method)

Parameters

v	
with_translation	
with_normalization	

Returns

vector

6.67.3.3 apply_to_vector()

```
glm::vec3 Transform::apply_to_vector (
             glm::vec3 & v )
```

Apply the Transform to a vector.

Parameters

v	

Returns

vector

6.67.3.4 apply_to_vector_of_point()

```
void Transform::apply_to_vector_of_point (
             std::vector< glm::vec3 > * points )
```

Apply the Transform to a list of point.

Parameters

points	

6.67.3.5 apply_to_vector_of_vec3()

```
void Transform::apply_to_vector_of_vec3 (
            std::vector< glm::vec3 > * vects,
            bool with_translation,
            bool with_normalization ) [protected]
```

Apply the Transform to a list of vec3 (General method)

Parameters

vects	
with_translation	
with_normalization	

6.67.3.6 apply_to_vector_of_vector()

```
void Transform::apply_to_vector_of_vector (
            std::vector< glm::vec3 > * vectors )
```

Apply the Transform to a list of vector.

Parameters

vectors	

6.67.3.7 apply_to_vector_of_versor()

```
void Transform::apply_to_vector_of_versor (
            std::vector< glm::vec3 > * versors )
```

Apply the Transform to a list of versor.

Parameters

versors	

6.67.3.8 apply_to_versor()

```
glm::vec3 Transform::apply_to_versor (
            glm::vec3 & v )
```

Apply the Transform to a versor.

Parameters

v	

Returns

versor

6.67.3.9 compute()

```
void Transform::compute ( )
```

Compute the matrix of the Transform.

6.67.3.10 compute_lerp_with_transform()

```
glm::mat4 Transform::compute_lerp_with_transform (
            Transform transf,
            float k )
```

Compute linear interpolation given an other Transform.

Parameters

transf	
k	

Returns

matrix

6.67.3.11 get_inverse()

```
glm::mat4 Transform::get_inverse ( )
```

Get the matrix that correspond at the inverse of the transform.

Returns

6.67.3.12 get_matrix()

```
const glm::mat4 & Transform::get_matrix ( )
```

Getter of the matrix.

Returns

matrix

6.67.3.13 get_rotation()

```
const glm::vec3 & Transform::get_rotation ( )
```

Getter of the rotation.

Returns

rotation

6.67.3.14 get_scale()

```
const glm::vec3 & Transform::get_scale ( )
```

Getter of the scale.

Returns

scale

6.67.3.15 get_translation()

```
const glm::vec3 & Transform::get_translation ( )
```

Getter of the translation.

Returns

translation

6.67.3.16 init()

```
void Transform::init (
             glm::vec3 translation = {0.0f, 0.0f, 0.0f},
             glm::vec3 rotation = {0.0f, 0.0f, 0.0f},
             glm::vec3 scale = {1.0f, 1.0f, 1.0f},
             int order_rotation = ORDER_YXZ )
```

Init method also used to reset te Transform.

Parameters

translation	
rotation	
scale	
order_rotation	

6.67.3.17 is_dirty()

```
TransformDirty * Transform::is_dirty ( ) const
```

Getter of the states dirty of the Transform.

Returns

is dirty

6.67.3.18 is_up_to_date()

```
bool Transform::is_up_to_date ( ) const
```

Getter of whether the Transform is up to date or no.

Returns

up to date

6.67.3.19 local_get_matrix()

```
glm::mat4 Transform::local_get_matrix ( )  [protected]
```

Compute the matrix of the Transform without saving it.

Returns

matrix

6.67.3.20 local_get_matrix_with_values()

```
glm::mat4 Transform::local_get_matrix_with_values (
             glm::vec3 tr,
             glm::vec3 rot,
             glm::vec3 sc,
             int order_rotation,
             bool inverse = false )  [static], [protected]
```

Compute a matrix with given values.

Parameters

tr	
rot	
sc	
order_rotation	

Returns

matrix

6.67.3.21 matrix_to_trs()

```
void Transform::matrix_to_trs (
            glm::mat4 matrix_to_decompose,
            glm::mat4 & t,
            glm::mat4 & r,
            glm::mat4 & s )  [static]
```

Decompose a matrix to 3 matrices(TRS)

Parameters

matrix_to_decompose	
t	
r	
s	

6.67.3.22 set_matrix()

```
void Transform::set_matrix (
            const glm::mat4 & new_matrix )
```

Setter of the matrix.

Parameters

new_matrix	

6.67.3.23 set_order_rotation()

```
void Transform::set_order_rotation (
            int order_rotation )
```

Setter of the order of rotation.

Parameters

order_rotation	

6.67.3.24 set_rotation()

```
void Transform::set_rotation (
             const glm::vec3 & new_rotation )
```

Setter of the rotation.

Parameters

new_rotation	

6.67.3.25 set_scale()

```
void Transform::set_scale (
             const glm::vec3 & new_scale )
```

Setter of the scale.

6.67.3.26 set_translation()

```
void Transform::set_translation (
             const glm::vec3 & new_translation )
```

Setter of the translation.

Parameters

new_translation	

6.67.3.27 set_uniform_scale()

```
void Transform::set_uniform_scale (
             float scale )
```

Setter of the scale with an uniform scale.

6.67.4 Friends And Related Function Documentation

6.67.4.1 operator"!=

```
bool operator!= (
            const Transform & trsf1,
            const Transform & trsf2 )  [friend]
```

Operator inequal.

Parameters

trsf1	
trsf2	

Returns

is_inequal

6.67.4.2 operator==

```
bool operator== (
            const Transform & trsf1,
            const Transform & trsf2 )  [friend]
```

Operator equal.

Parameters

trsf1	
trsf2	

Returns

is_equal

6.67.5 Member Data Documentation

6.67.5.1 m_dirty

```
TransformDirty* Transform::m_dirty {}  [protected]
```

6.67.5.2 m_matrix

```
glm::mat4 Transform::m_matrix  [protected]
```

6.67.5.3 m_order_rotation

```
int Transform::m_order_rotation {}  [protected]
```

6.67.5.4 m_rot

```
glm::vec3 Transform::m_rot  [protected]
```

6.67.5.5 m_scale

```
glm::vec3 Transform::m_scale  [protected]
```

6.67.5.6 m_translate

```
glm::vec3 Transform::m_translate  [protected]
```

6.67.5.7 m_up_to_date

```
bool Transform::m_up_to_date {}  [protected]
```

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

- src/utils/Transform.hpp
- src/utils/Transform.cpp

6.68 TransformDirty Struct Reference

State of a Transform.

```
#include <Transform.hpp>
```

Public Member Functions

- TransformDirty (bool dirty)
- bool has_dirty () const
- void logic_or (TransformDirty dirty)
- void logic_and (TransformDirty dirty)
- void reset (bool dirty)

Public Attributes

- bool translation {}
- bool rotation {}
- bool scale {}
- bool matrix {}

6.68.1 Detailed Description

State of a Transform.

6.68.2 Constructor & Destructor Documentation

6.68.2.1 TransformDirty()

```
TransformDirty::TransformDirty (
            bool dirty )  [explicit]
```

6.68.3 Member Function Documentation

6.68.3.1 has_dirty()

```
bool TransformDirty::has_dirty ( ) const
```

6.68.3.2 logic_and()

```
void TransformDirty::logic_and (
            TransformDirty dirty )
```

6.68.3.3 logic_or()

```
void TransformDirty::logic_or (
             TransformDirty dirty )
```

6.68.3.4 reset()

```
void TransformDirty::reset (
             bool dirty )
```

6.68.4 Member Data Documentation

6.68.4.1 matrix

```
bool TransformDirty::matrix {}
```

6.68.4.2 rotation

```
bool TransformDirty::rotation {}
```

6.68.4.3 scale

```
bool TransformDirty::scale {}
```

6.68.4.4 translation

```
bool TransformDirty::translation {}
```

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

- src/utils/Transform.hpp
- src/utils/Transform.cpp

6.69 shader::VAODataManager Class Reference

Manager of VBOs and EBO.

```
#include <VAODataManager.hpp>
```

Static Public Member Functions

- static void generate_vao (GLuint ∗vao_id)

  Generate a new VAO.
- static void generate_bo (GLuint ∗bo_id)

  Generate a new VBO.
- static void bind_vao (GLuint vao_id)

  Bind the VAO with the given id.
- static void enable_attrib_vbo (GLuint index_vbo, GLuint vbo_id, GLint size_data, GLboolean normalized)

  Enable the pointer for a VBO.
- static void delete_bo (GLuint bo_id)

  Delete a BO.
- static void delete_vao (GLuint vao_id)

  Delete a VAO.
- template<typename T >
  static void fill_bo (GLenum buffer_type, GLuint vbo_id, std::vector< T > datas)

  Fill a buffer with datas.
- static void disable_attrib_vbo (GLuint index_vbo)

  Disable the pointer for a VBO.
- static void draw (GLuint ebo_id, long nb_indices)

  Draw the EBO with the given id.
- static void draw_verticies_debug (std::vector< glm::vec3 > verticies)

  Draw a list of verticies as a debug rendering.

Static Public Attributes

- const static int ID_VERTEX_BUFFER = 0
- const static int ID_NORMAL_BUFFER = 1
- const static int ID_UV_BUFFER = 2

6.69.1 Detailed Description

Manager of VBOs and EBO.

6.69.2 Member Function Documentation

6.69.2.1 bind_vao()

```
void VAODataManager::bind_vao (
            GLuint vao_id ) [static]
```

Bind the VAO with the given id.

Parameters

vao← _id	

6.69.2.2 delete_bo()

```
void VAODataManager::delete_bo (
             GLuint bo_id )  [static]
```

Delete a BO.

Parameters

bo← _id	

6.69.2.3 delete_vao()

```
void VAODataManager::delete_vao (
             GLuint vao_id )  [static]
```

Delete a VAO.

Parameters

vao← _id	

6.69.2.4 disable_attrib_vbo()

```
void VAODataManager::disable_attrib_vbo (
             GLuint index_vbo )  [static]
```

Disable the pointer for a VBO.

Parameters

index_vbo	

6.69.2.5 draw()

```
void VAODataManager::draw (
            GLuint ebo_id,
            long nb_indices )  [static]
```

Draw the EBO with the given id.

Parameters

ebo_id	
nb_indices	

6.69.2.6 draw_verticies_debug()

```
void VAODataManager::draw_verticies_debug (
            std::vector< glm::vec3 > verticies )  [static]
```

Draw a list of verticies as a debug rendering.

Parameters

verticies	

6.69.2.7 enable_attrib_vbo()

```
void VAODataManager::enable_attrib_vbo (
            GLuint index_vbo,
            GLuint vbo_id,
            GLint size_data,
            GLboolean normalized )  [static]
```

Enable the pointer for a VBO.

Parameters

index_vbo	
vbo_id	
size_data	
normalized	

6.69.2.8 fill_bo()

```
template<typename T >
template void VAODataManager::fill_bo< unsigned short int > (
            GLenum buffer_type,
            GLuint vbo_id,
            std::vector< T > datas )  [static]
```

Fill a buffer with datas.

Template Parameters

T	

Parameters

buffer_type	
vbo_id	
datas	

6.69.2.9 generate_bo()

```
void VAODataManager::generate_bo (
            GLuint * bo_id )  [static]
```

Generate a new VBO.

Parameters

bo↩ _id	

6.69.2.10 generate_vao()

```
void VAODataManager::generate_vao (
            GLuint * vao_id )  [static]
```

Generate a new VAO.

Parameters

vao↩ _id	

6.69.3 Member Data Documentation

6.69.3.1 ID_NORMAL_BUFFER

```
const static int shader::VAODataManager::ID_NORMAL_BUFFER = 1  [static]
```

6.69.3.2 ID_UV_BUFFER

```
const static int shader::VAODataManager::ID_UV_BUFFER = 2  [static]
```

6.69.3.3 ID_VERTEX_BUFFER

```
const static int shader::VAODataManager::ID_VERTEX_BUFFER = 0  [static]
```

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

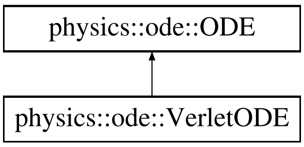
- src/shader/VAODataManager.hpp
- src/shader/VAODataManager.cpp

6.70 physics::ode::VerletODE Class Reference

Verlet Ordinary Differential Equation.

```
#include <VerletODE.hpp>
```

Inheritance diagram for physics::ode::VerletODE:



Public Member Functions

- VerletODE ()
- void update (RigidBodyVolume ∗rbv, float delta_time) override

    Update position and rotation of a RigidBodyVolume with an ODE equation.

Additional Inherited Members

6.70.1 Detailed Description

Verlet Ordinary Differential Equation.

6.70.2 Constructor & Destructor Documentation

6.70.2.1 VerletODE()

```
VerletODE::VerletODE ( )
```

6.70.3 Member Function Documentation

6.70.3.1 update()

```
void VerletODE::update (
            RigidBodyVolume * rbv,
            float delta_time )  [override], [virtual]
```

Update position and rotation of a RigidBodyVolume with an ODE equation.

Parameters

rbv	
delta_time	

Implements physics::ode::ODE.

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

- src/physics/ode/VerletODE.hpp
- src/physics/ode/VerletODE.cpp

Chapter 7

File Documentation

7.1 main/main.cpp File Reference

```
#include <cstdio>
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <src/scene/SceneLand.hpp>
#include <src/scene/SolarSystem.hpp>
#include <src/scene/BounceOBBScene.hpp>
#include <src/scene/BounceAABBScene.hpp>
#include <src/scene/BounceSphereBBScene.hpp>
#include <src/scene/LabScene.hpp>
#include <src/scene/ShadowedScene.hpp>
#include <src/utils/printer.hpp>
```

Functions

- void window_size_callback (GLFWwindow ∗window, int width, int height)
- int main ()

Variables

- GLFWwindow ∗ window

7.1.1 Function Documentation

7.1.1.1 main()

```
int main ( )
```

7.1.1.2 window_size_callback()

```
void window_size_callback (
            GLFWwindow * window,
            int width,
            int height )
```

7.1.2 Variable Documentation

7.1.2.1 window

```
GLFWwindow* window
```

7.2 src/game_element/ButtonElement.cpp File Reference

```
#include "ButtonElement.hpp"
```

7.3 src/game_element/ButtonElement.hpp File Reference

```
#include "../scene_graph/NodeGameSG.hpp"
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
#include <src/physics/rigid_body_behavior/MoveDoorBehavior.hpp>
#include <src/physics/rigid_body_behavior/SwitchColorBehavior.hpp>
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
#include "DoorElement.hpp"
```

Classes

- class ButtonElement

7.4 src/game_element/Character.cpp File Reference

```
#include "Character.hpp"
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
```

7.5 src/game_element/Character.hpp File Reference

```
#include "src/scene_graph/RootSG.hpp"
#include "src/physics/force/GravityForce.hpp"
#include "src/utils/meshloader.hpp"
#include "src/material/MaterialColor.hpp"
#include "src/MouseView.hpp"
#include "src/shader/MainShaders.hpp"
```

Classes

- class Character

7.6 src/game_element/DoorElement.cpp File Reference

```
#include "DoorElement.hpp"
```

7.7 src/game_element/DoorElement.hpp File Reference

```
#include "../scene_graph/NodeGameSG.hpp"
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
```

Classes

- class DoorElement

7.8 src/light/DirectionLight.cpp File Reference

```
#include "DirectionLight.hpp"
#include <utility>
```

7.9 src/light/DirectionLight.hpp File Reference

```
#include "src/light/Light.hpp"
#include "src/light/light_behavior/DirectionLightBehavior.hpp"
```

Classes

- class light::DirectionLight

  Light with a direction.

Namespaces

- light

7.10 src/light/Light.cpp File Reference

```
#include "Light.hpp"
#include <utility>
```

7.11 src/light/Light.hpp File Reference

```
#include <GL/glew.h>
#include <cstdio>
#include <cstdlib>
#include <string>
#include <vector>
#include <glm/glm.hpp>
#include "LightShader.hpp"
#include "src/light/light_behavior/LightBehavior.hpp"
```

Classes

- class light::Light

  Base Light (Abstract)

Namespaces

- light
- light::behavior

7.12 src/light/light_behavior/DirectionLightBehavior.cpp File Reference

```
#include "DirectionLightBehavior.hpp"
```

7.13 src/light/light_behavior/DirectionLightBehavior.hpp File Reference

```
#include "LightBehavior.hpp"
#include "src/light/Light.hpp"
```

Classes

- class light::behavior::DirectionLightBehavior

  Behavior of a directed Light.

Namespaces

- light
- light::behavior

7.14 src/light/light_behavior/LightBehavior.cpp File Reference

```
#include "LightBehavior.hpp"
```

7.15 src/light/light_behavior/LightBehavior.hpp File Reference

```
#include "src/light/LightShader.hpp"
#include "src/scene_graph/NodeGameSG.hpp"
```

Classes

- class light::behavior::LightBehavior

  Behavior of a Light (Abstract)

Namespaces

- scene_graph
- light
- light::behavior

7.16 src/light/light_behavior/PositionLightBehavior.cpp File Reference

```
#include "PositionLightBehavior.hpp"
```

7.17 src/light/light_behavior/PositionLightBehavior.hpp File Reference

```
#include "LightBehavior.hpp"
#include "src/light/Light.hpp"
```

Classes

- class light::behavior::PositionLightBehavior

  Behavior of a positionned Light.

Namespaces

- light
- light::behavior

7.18 src/light/light_behavior/SpotLightBehavior.cpp File Reference

```
#include "SpotLightBehavior.hpp"
#include <src/utils/printer.hpp>
```

7.19 src/light/light_behavior/SpotLightBehavior.hpp File Reference

```
#include <src/shader/ShadowMap.hpp>
#include "src/light/light_behavior/LightBehavior.hpp"
#include "src/light/Light.hpp"
```

Classes

- class light::behavior::SpotLightBehavior

  Behavior of a SpotLight.

Namespaces

- light
- light::behavior

7.20 src/light/LightShader.cpp File Reference

```
#include "src/light/LightShader.hpp"
```

7.21 src/light/LightShader.hpp File Reference

```
#include <src/shader/ShadowMap.hpp>
#include "src/shader/Shaders.hpp"
```

Classes

- struct light::LightInfo

  Light Information.
- struct light::LightShader

  Light in the shader based on data retrieved from Light objects

Namespaces

- light

7.22 src/light/PositionLight.cpp File Reference

```
#include "PositionLight.hpp"
#include <utility>
```

7.23 src/light/PositionLight.hpp File Reference

```
#include "Light.hpp"
#include "src/light/light_behavior/PositionLightBehavior.hpp"
```

Classes

- class light::PositionLight

  Light with a position and whose intensity can be attenuated.

Namespaces

- light

7.24 src/light/SpotLight.cpp File Reference

```
#include "SpotLight.hpp"
```

7.25 src/light/SpotLight.hpp File Reference

```
#include "Light.hpp"
#include "src/light/light_behavior/SpotLightBehavior.hpp"
#include "src/light/light_behavior/PositionLightBehavior.hpp"
#include "src/light/light_behavior/DirectionLightBehavior.hpp"
```

Classes

- class light::SpotLight

  Light with a direction, a position and angles of diffusion.

Namespaces

- light

7.26 src/material/Material.cpp File Reference

```
#include "Material.hpp"
```

7.27 src/material/Material.hpp File Reference

```
#include <GL/glew.h>
#include <cstdio>
#include <cstdlib>
#include <string>
#include <utility>
#include <iostream>
#include <src/shader/Shaders.hpp>
```

Classes

- class material::Material

  Base Material (Abstract)

Namespaces

- material

7.28 src/material/MaterialColor.cpp File Reference

```
#include "MaterialColor.hpp"
#include <utility>
```

7.29 src/material/MaterialColor.hpp File Reference

```
#include "Material.hpp"
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
```

Classes

- class material::MaterialColor

  Material with colors for the ambient, diffuse and specular components.

Namespaces

- material

7.30 src/material/MaterialTexture.cpp File Reference

```
#include "MaterialTexture.hpp"
#include <utility>
```

7.31 src/material/MaterialTexture.hpp File Reference

```
#include "Material.hpp"
#include <src/utils/texture.hpp>
```

Classes

- class material::MaterialTexture

  Material with texture for the diffuse and specular components.

Namespaces

- material

7.32 src/mesh/LODMesh.cpp File Reference

```
#include "LODMesh.hpp"
```

7.33 src/mesh/LODMesh.hpp File Reference

```
#include "Mesh.hpp"
```

Classes

- class mesh::LODMesh

  Mesh with Level of Details.

Namespaces

- mesh

7.34 src/mesh/Mesh.cpp File Reference

```
#include <src/physics/bounding_box/AABB.hpp>
#include <src/physics/bounding_box/BBFactory.hpp>
#include "Mesh.hpp"
#include <src/utils/printer.hpp>
```

7.35 src/mesh/Mesh.hpp File Reference

```
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <iostream>
#include <GL/glew.h>
#include <vector>
#include <cmath>
#include <src/utils/Transform.hpp>
#include <src/shader/VAODataManager.hpp>
#include <src/physics/bounding_box/BBFactory.hpp>
#include <src/physics/bounding_box/BoundingBox.hpp>
#include "src/utils/objloader.hpp"
```

Classes

- struct mesh::MeshData

    Datas of a Mesh.
- class mesh::Mesh

    Base Mesh.

Namespaces

- mesh

7.36 src/MouseView.cpp File Reference

```
#include "MouseView.hpp"
```

7.37 src/MouseView.hpp File Reference

```
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <iostream>
```

Classes

- class MouseView

7.38 src/physics/bounding_box/AABB.cpp File Reference

```
#include "AABB.hpp"
#include "OBB.hpp"
#include "SphereBB.hpp"
#include "src/physics/Collision.hpp"
#include <src/utils/printer.hpp>
```

7.39 src/physics/bounding_box/AABB.hpp File Reference

```
#include "RCBB.hpp"
```

Classes

- class physics::bounding_box::AABB

  BoundingBox aligned on axis.

Namespaces

- physics
- physics::bounding_box

7.40 src/physics/bounding_box/BBFactory.cpp File Reference

```
#include "BBFactory.hpp"
#include "AABB.hpp"
#include "OBB.hpp"
#include "SphereBB.hpp"
```

7.41 src/physics/bounding_box/BBFactory.hpp File Reference

```
#include "BoundingBox.hpp"
```

Classes

- class physics::bounding_box::BBFactory

  Factory of BoundingBox.

Namespaces

- physics
- physics::bounding_box

7.42 src/physics/bounding_box/BoundingBox.cpp File Reference

```
#include <iostream>
#include "BoundingBox.hpp"
#include "AABB.hpp"
```

7.43 src/physics/bounding_box/BoundingBox.hpp File Reference

```
#include <glm/glm.hpp>
#include <vector>
#include <src/utils/Geometry3D.hpp>
#include <src/utils/Transform.hpp>
```

Classes

- class physics::bounding_box::BoundingBox

  Represent a bounding box (Abstract)

Namespaces

- utils
- physics
- physics::bounding_box

Enumerations

- enum physics::bounding_box::BB_TYPE { physics::bounding_box::AABB_TYPE, physics::bounding_box::OBB_TYPE, physics::bounding_box::SPHEREBB_TYPE }

  Enum of the different types of BoundingBox.

7.44 src/physics/bounding_box/OBB.cpp File Reference

```
#include <iostream>
#include "OBB.hpp"
#include "AABB.hpp"
#include "SphereBB.hpp"
#include "src/physics/Collision.hpp"
#include <src/utils/printer.hpp>
#include <src/utils/Transform.hpp>
```

7.45 src/physics/bounding_box/OBB.hpp File Reference

```
#include "RCBB.hpp"
```

Classes

- class physics::bounding_box::OBB

  BoundingBox oriented.

Namespaces

- physics
- physics::bounding_box

7.46 src/physics/bounding_box/RCBB.cpp File Reference

```
#include "RCBB.hpp"
#include "AABB.hpp"
#include "SphereBB.hpp"
#include <glm/glm.hpp>
#include <iostream>
#include <src/physics/Collision.hpp>
#include <src/utils/printer.hpp>
```

7.47 src/physics/bounding_box/RCBB.hpp File Reference

```
#include "BoundingBox.hpp"
#include <cfloat>
```

Classes

- class physics::bounding_box::RCBB

  Rectangle Cuboid BoundingBox (Abstract)

Namespaces

- utils
- physics
- physics::bounding_box

7.48 src/physics/bounding_box/SphereBB.cpp File Reference

```
#include <iostream>
#include <src/utils/Transform.hpp>
#include "SphereBB.hpp"
#include "AABB.hpp"
#include "OBB.hpp"
#include <src/utils/printer.hpp>
```

7.49 src/physics/bounding_box/SphereBB.hpp File Reference

```
#include "BoundingBox.hpp"
#include <cfloat>
#include <src/physics/Collision.hpp>
```

Classes

- class physics::bounding_box::SphereBB

  Sphere BoundingBox.

Namespaces

- physics
- physics::bounding_box

7.50 src/physics/Collision.cpp File Reference

```
#include <sstream>
#include "Collision.hpp"
```

7.51 src/physics/Collision.hpp File Reference

```
#include <glm/glm.hpp>
#include <vector>
#include <string>
#include "src/utils/Geometry3D.hpp"
```

Classes

- struct physics::Collision

  Represents a collision between 2 BoundingBox.

Namespaces

- physics

7.52 src/physics/force/Force.cpp File Reference

```
#include "Force.hpp"
```

7.53 src/physics/force/Force.hpp File Reference

```
#include "src/physics/RigidBodyVolume.hpp"
```

Classes

- class physics::force::Force

    Represent a Force to apply on a RigidBodyVolume (Abstract)

Namespaces

- physics
- physics::force

7.54 src/physics/force/GravityForce.cpp File Reference

```
#include "GravityForce.hpp"
#include "src/physics/rigid_body_behavior/MovementBehavior.hpp"
```

7.55 src/physics/force/GravityForce.hpp File Reference

```
#include "Force.hpp"
#include "src/physics/RigidBodyVolume.hpp"
```

Classes

- class physics::force::GravityForce

    Force of the gravity.

Namespaces

- physics
- physics::force

7.56 src/physics/ode/EulerODE.cpp File Reference

```
#include "EulerODE.hpp"
#include <src/physics/RigidBodyVolume.hpp>
#include "src/utils/printer.hpp"
#include "src/physics/rigid_body_behavior/MovementBehavior.hpp"
```

7.57 src/physics/ode/EulerODE.hpp File Reference

```
#include "ODE.hpp"
```

Classes

- class physics::ode::EulerODE

  Euler Ordinary Differential Equation.

Namespaces

- physics
- physics::ode

7.58 src/physics/ode/ODE.cpp File Reference

```
#include "ODE.hpp"
#include <src/physics/RigidBodyVolume.hpp>
```

7.59 src/physics/ode/ODE.hpp File Reference

```
#include <glm/glm.hpp>
```

Classes

- class physics::ode::ODE

  Ordinary Differential Equation (Abstract)

Namespaces

- physics
- physics::ode

Enumerations

- enum physics::ode::ODE_TYPE { physics::ode::EULER_TYPE, physics::ode::VERLET_TYPE, physics::ode::RK4_TYPE }

  Enum of the different types of ODE.

7.60 src/physics/ode/ODEFactory.cpp File Reference

```
#include "ODEFactory.hpp"
#include "EulerODE.hpp"
#include "VerletODE.hpp"
#include "RungeKutta4ODE.hpp"
```

7.61 src/physics/ode/ODEFactory.hpp File Reference

```
#include "ODE.hpp"
```

Classes

- class physics::ode::ODEFactory

  Factory of ODE.

Namespaces

- physics
- physics::ode

7.62 src/physics/ode/RungeKutta4ODE.cpp File Reference

```
#include "RungeKutta4ODE.hpp"
#include <src/physics/RigidBodyVolume.hpp>
#include <src/utils/printer.hpp>
#include "src/physics/rigid_body_behavior/MovementBehavior.hpp"
```

7.63 src/physics/ode/RungeKutta4ODE.hpp File Reference

```
#include "ODE.hpp"
```

Classes

- class physics::ode::RungeKutta4ODE

  Runge Kutta Ordinary Differential Equation.

Namespaces

- physics
- physics::ode

7.64 src/physics/ode/VerletODE.cpp File Reference

```
#include "VerletODE.hpp"
#include <src/physics/RigidBodyVolume.hpp>
#include "src/physics/rigid_body_behavior/MovementBehavior.hpp"
```

7.65 src/physics/ode/VerletODE.hpp File Reference

```
#include "ODE.hpp"
```

Classes

- class physics::ode::VerletODE

  Verlet Ordinary Differential Equation.

Namespaces

- physics
- physics::ode

7.66 src/physics/PhysicsSystem.cpp File Reference

```
#include <algorithm>
#include "PhysicsSystem.hpp"
#include "RigidBodyVolume.hpp"
#include "src/physics/ode/ODEFactory.hpp"
```

7.67 src/physics/PhysicsSystem.hpp File Reference

```
#include <vector>
#include "Collision.hpp"
#include "src/physics/ode/ODE.hpp"
#include "src/scene_graph/NodeGameSG.hpp"
```

Classes

- class physics::PhysicsSystem

  Manage a whole physic system with RigidBody.

Namespaces

- scene_graph
- physics

7.68 src/physics/rigid_body_behavior/MoveDoorBehavior.cpp File Reference

```
#include "MoveDoorBehavior.hpp"
```

7.69 src/physics/rigid_body_behavior/MoveDoorBehavior.hpp File Reference

```
#include "RigidBodyBehavior.hpp"
#include <src/physics/PhysicsSystem.hpp>
#include <src/physics/force/Force.hpp>
#include "src/game_element/DoorElement.hpp"
```

Classes

- class physics::rigid_body_behavior::MoveDoorBehavior

Namespaces

- physics
- physics::rigid_body_behavior

7.70 src/physics/rigid_body_behavior/MovementBehavior.cpp File Reference

```
#include "MovementBehavior.hpp"
```

7.71 src/physics/rigid_body_behavior/MovementBehavior.hpp File Reference

```
#include "RigidBodyBehavior.hpp"
#include <src/physics/PhysicsSystem.hpp>
#include <src/physics/force/Force.hpp>
```

Classes

- class physics::rigid_body_behavior::MovementBehavior

Namespaces

- physics
- physics::force
- physics::rigid_body_behavior

7.72 src/physics/rigid_body_behavior/RigidBodyBehavior.cpp File Reference

```
#include "RigidBodyBehavior.hpp"
```

7.73 src/physics/rigid_body_behavior/RigidBodyBehavior.hpp File Reference

```
#include <src/physics/RigidBodyVolume.hpp>
```

Classes

- class physics::rigid_body_behavior::RigidBodyBehavior

Namespaces

- physics
- physics::ode
- physics::rigid_body_behavior

Enumerations

- enum physics::rigid_body_behavior::RigidBodyBehavior_TYPE { physics::rigid_body_behavior::MovementBehavior_TYPE, physics::rigid_body_behavior::SwitchColor_TYPE, physics::rigid_body_behavior::MoveDoor_TYPE }

7.74 src/physics/rigid_body_behavior/SwitchColorBehavior.cpp File Reference

```
#include "SwitchColorBehavior.hpp"
```

7.75 src/physics/rigid_body_behavior/SwitchColorBehavior.hpp File Reference

```
#include "RigidBodyBehavior.hpp"
#include <src/physics/PhysicsSystem.hpp>
#include <src/physics/force/Force.hpp>
```

Classes

- class physics::rigid_body_behavior::SwitchColorBehavior

Namespaces

- physics
- physics::rigid_body_behavior

7.76 src/physics/RigidBodyVolume.cpp File Reference

```
#include "RigidBodyVolume.hpp"
#include "src/physics/Collision.hpp"
#include "src/physics/ode/ODEFactory.hpp"
#include "src/utils/printer.hpp"
#include <glm/gtx/matrix_decompose.hpp>
#include "src/physics/rigid_body_behavior/MovementBehavior.hpp"
```

7.77 src/physics/RigidBodyVolume.hpp File Reference

```
#include "src/physics/bounding_box/BoundingBox.hpp"
#include "src/physics/bounding_box/SphereBB.hpp"
#include "src/physics/bounding_box/OBB.hpp"
#include "src/scene_graph/NodeGameSG.hpp"
#include "src/physics/force/Force.hpp"
#include "src/physics/ode/ODE.hpp"
```

Classes

- class physics::RigidBodyVolume

  Represents a rigid body with a volume (BoundingBox)

Namespaces

- scene_graph
- physics
- physics::force
- physics::rigid_body_behavior

7.78 src/scene/BounceAABBScene.cpp File Reference

```
#include <src/physics/force/GravityForce.hpp>
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
#include "BounceAABBScene.hpp"
```

7.79 src/scene/BounceAABBScene.hpp File Reference

```
#include "Scene.hpp"
#include "src/mesh/LODMesh.hpp"
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
#include "src/light/DirectionLight.hpp"
#include "src/utils/objloader.hpp"
#include <src/light/PositionLight.hpp>
#include <src/scene_graph/NodeOnTopSG.hpp>
#include <src/light/SpotLight.hpp>
```

Classes

- class scene::BounceAABBScene

  Scene with Collisions.

Namespaces

- scene

7.80 src/scene/BounceOBBScene.cpp File Reference

```
#include <src/physics/force/GravityForce.hpp>
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
#include "BounceOBBScene.hpp"
```

7.81 src/scene/BounceOBBScene.hpp File Reference

```
#include "Scene.hpp"
#include "src/mesh/LODMesh.hpp"
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
#include "src/light/DirectionLight.hpp"
#include "src/utils/objloader.hpp"
#include <src/light/PositionLight.hpp>
#include <src/scene_graph/NodeOnTopSG.hpp>
#include <src/light/SpotLight.hpp>
```

Classes

- class scene::BounceOBBScene

  Scene with Collisions.

Namespaces

- scene

7.82 src/scene/BounceSphereBBScene.cpp File Reference

```
#include <src/physics/force/GravityForce.hpp>
#include "BounceSphereBBScene.hpp"
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
```

7.83 src/scene/BounceSphereBBScene.hpp File Reference

```
#include "Scene.hpp"
#include "src/mesh/LODMesh.hpp"
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
#include "src/light/DirectionLight.hpp"
#include "src/utils/objloader.hpp"
#include <src/light/PositionLight.hpp>
#include <src/scene_graph/NodeOnTopSG.hpp>
#include <src/light/SpotLight.hpp>
```

Classes

- class scene::BounceSphereBBScene

  Scene with Collisions.

Namespaces

- scene

7.84 src/scene/LabScene.cpp File Reference

```
#include <src/physics/force/GravityForce.hpp>
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
#include <src/physics/rigid_body_behavior/SwitchColorBehavior.hpp>
#include "LabScene.hpp"
```

7.85 src/scene/LabScene.hpp File Reference

```
#include "Scene.hpp"
#include "src/mesh/LODMesh.hpp"
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
#include "src/light/DirectionLight.hpp"
#include "src/utils/objloader.hpp"
#include <src/light/PositionLight.hpp>
#include <src/scene_graph/NodeOnTopSG.hpp>
#include <src/light/SpotLight.hpp>
#include "src/game_element/Character.hpp"
#include "src/game_element/DoorElement.hpp"
#include "src/game_element/ButtonElement.hpp"
```

Classes

- class scene::LabScene

  Scene with Collisions.

Namespaces

- scene

7.86 src/scene/Scene.cpp File Reference

```
#include <src/shader/ShadowMap.hpp>
#include "Scene.hpp"
```

7.87 src/scene/Scene.hpp File Reference

```
#include "glm/ext.hpp"
#include <src/scene_graph/RootSG.hpp>
#include <src/scene_graph/NodeGameSG.hpp>
#include <src/light/Light.hpp>
#include <src/material/Material.hpp>
#include <src/utils/meshloader.hpp>
#include <src/utils/texture.hpp>
#include <src/physics/PhysicsSystem.hpp>
#include "src/shader/MainShaders.hpp"
```

Classes

- class scene::Scene

  Base Scene (Abstract)

Namespaces

- scene

7.88 src/scene/SceneLand.cpp File Reference

```
#include <src/light/PositionLight.hpp>
#include <src/scene_graph/NodeOnTopSG.hpp>
#include <src/light/SpotLight.hpp>
#include "SceneLand.hpp"
```

7.89 src/scene/SceneLand.hpp File Reference

```
#include "Scene.hpp"
#include "../mesh/LODMesh.hpp"
#include "../material/MaterialTexture.hpp"
#include "../material/MaterialColor.hpp"
#include "src/light/DirectionLight.hpp"
#include "../utils/objloader.hpp"
```

Classes

- class scene::SceneLand

  Scene with a land and a ball on it.

Namespaces

- scene

7.90 src/scene/ShadowedScene.cpp File Reference

```
#include "ShadowedScene.hpp"
#include <src/physics/force/GravityForce.hpp>
#include <src/physics/rigid_body_behavior/MovementBehavior.hpp>
```

7.91 src/scene/ShadowedScene.hpp File Reference

```
#include "Scene.hpp"
#include "src/mesh/LODMesh.hpp"
#include "src/material/MaterialTexture.hpp"
#include "src/material/MaterialColor.hpp"
#include "src/light/DirectionLight.hpp"
#include "src/utils/objloader.hpp"
#include <src/light/PositionLight.hpp>
#include <src/scene_graph/NodeOnTopSG.hpp>
#include <src/light/SpotLight.hpp>
```

Classes

- class scene::ShadowedScene

  Scene with Collisions.

Namespaces

- scene

7.92 src/scene/SolarSystem.cpp File Reference

```
#include "SolarSystem.hpp"
```

7.93 src/scene/SolarSystem.hpp File Reference

```
#include "Scene.hpp"
#include <src/material/MaterialColor.hpp>
#include <src/material/MaterialTexture.hpp>
#include <src/light/PositionLight.hpp>
#include <src/mesh/LODMesh.hpp>
```

Classes

- class scene::SolarSystem

  Scene with the sun, the earth, the moon end the sky.

Namespaces

- scene

7.94 src/scene_graph/ElementSG.cpp File Reference

```
#include "ElementSG.hpp"
#include "NodeSG.hpp"
#include <utility>
#include <src/shader/Shaders.hpp>
```

7.95 src/scene_graph/ElementSG.hpp File Reference

```
#include <map>
#include <src/shader/VAODataManager.hpp>
#include <src/shader/MainShaders.hpp>
#include "src/utils/Transform.hpp"
#include <utility>
```

Classes

- class scene_graph::ElementSG

  Base Element of the scene graph (Abstract)

Namespaces

- shader
- scene_graph

7.96 src/scene_graph/NodeGameSG.cpp File Reference

```
#include "NodeGameSG.hpp"
#include <src/utils/printer.hpp>
#include <utility>
```

7.97 src/scene_graph/NodeGameSG.hpp File Reference

```
#include "NodeSG.hpp"
#include <src/material/Material.hpp>
#include <src/light/Light.hpp>
#include <src/physics/bounding_box/BoundingBox.hpp>
#include <src/physics/bounding_box/AABB.hpp>
#include "src/mesh/Mesh.hpp"
#include <src/physics/RigidBodyVolume.hpp>
```

Classes

- class scene_graph::NodeGameSG

  NodeSG that can have Meshes, light and camera.

Namespaces

- light
- scene_graph

Macros

- #define NODE_INIT_POSITION glm::vec3(0, 0, 0)
- #define NODE_INIT_FORWARD glm::vec3(0, 0, -1)
- #define NODE_INIT_UP glm::vec3(0, 1, 0)

7.97.1 Macro Definition Documentation

7.97.1.1 NODE_INIT_FORWARD

```
#define NODE_INIT_FORWARD glm::vec3(0, 0, -1)
```

7.97.1.2 NODE_INIT_POSITION

```
#define NODE_INIT_POSITION glm::vec3(0, 0, 0)
```

7.97.1.3 NODE_INIT_UP

```
#define NODE_INIT_UP glm::vec3(0, 1, 0)
```

7.98 src/scene_graph/NodeOnTopSG.cpp File Reference

```
#include "NodeOnTopSG.hpp"
```

7.99 src/scene_graph/NodeOnTopSG.hpp File Reference

```
#include "NodeGameSG.hpp"
```

Classes

- class scene_graph::NodeOnTopSG

  NodeGameSG on top of another.

Namespaces

- scene_graph

7.100 src/scene_graph/NodeSG.cpp File Reference

```
#include "NodeSG.hpp"
#include <utility>
#include <src/physics/bounding_box/BBFactory.hpp>
#include <src/physics/bounding_box/AABB.hpp>
#include <src/utils/printer.hpp>
```

7.101 src/scene_graph/NodeSG.hpp File Reference

```
#include <vector>
#include "ElementSG.hpp"
#include "src/utils/Transform.hpp"
#include "src/shader/Shaders.hpp"
```

Classes

- class scene_graph::NodeSG

  Base Node of the scene graph (Abstract)

Namespaces

- scene_graph

7.102 src/scene_graph/RootSG.cpp File Reference

```
#include "RootSG.hpp"
#include <utility>
```

7.103 src/scene_graph/RootSG.hpp File Reference

```
#include "NodeSG.hpp"
```

Classes

- class scene_graph::RootSG

  Root of the scene graph.

Namespaces

- scene_graph

7.104 src/shader/MainShaders.cpp File Reference

```
#include "MainShaders.hpp"
```

7.105 src/shader/MainShaders.hpp File Reference

```
#include "Shaders.hpp"
#include "ShadowMapShaders.hpp"
```

Classes

- class shader::MainShaders

Namespaces

- shader

7.106 src/shader/Shaders.cpp File Reference

```
#include "Shaders.hpp"
```

7.107 src/shader/Shaders.hpp File Reference

```
#include <cstdio>
#include <string>
#include <vector>
#include <iostream>
#include <fstream>
#include <algorithm>
#include <sstream>
#include "GL/glew.h"
#include "TextureManager.hpp"
#include <src/shader/ShadersDataManager.hpp>
```

Classes

- struct shader::glsl_bool

  Vec3 used in shader.
- struct shader::glsl_int

  Vec3 used in shader.
- struct shader::glsl_vec3

  Vec3 used in shader.
- struct shader::glsl_mat4

  Mat4 used in shader.
- class shader::Shaders

  Group the VAOManager, the ShadersDataManager and the TextureManager.

Namespaces

- shader

7.108 src/shader/ShadersDataManager.cpp File Reference

```
#include "ShadersDataManager.hpp"
#include <src/material/Material.hpp>
#include <src/light/Light.hpp>
#include <src/shader/ShadowMapShaders.hpp>
```

7.109 src/shader/ShadersDataManager.hpp File Reference

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <map>
#include <vector>
```

Classes

- class shader::ShadersDataManager

  Manager of the locations in shaders.

Namespaces

- light
- shader

7.110 src/shader/ShadowMap.cpp File Reference

```
#include "ShadowMap.hpp"
#include "ShadowMapShaders.hpp"
```

7.111 src/shader/ShadowMap.hpp File Reference

```
#include "Shaders.hpp"
```

Classes

- class shader::ShadowMap

  Represents a Shadow Map.

Namespaces

- shader

7.112 src/shader/ShadowMapShaders.cpp File Reference

```
#include "ShadowMapShaders.hpp"
```

7.113 src/shader/ShadowMapShaders.hpp File Reference

```
#include "ShadersDataManager.hpp"
#include "Shaders.hpp"
#include "src/light/LightShader.hpp"
#include <iostream>
```

Classes

- class shader::ShadowMapShaders

  Manager of the Shadow Mapping Shaders.

Namespaces

- shader

7.114 src/shader/TextureManager.cpp File Reference

```
#include <iostream>
#include "TextureManager.hpp"
```

7.115 src/shader/TextureManager.hpp File Reference

```
#include <GL/glew.h>
#include "ShadersDataManager.hpp"
#include <src/utils/texture.hpp>
```

Classes

- class shader::TextureManager

  Manager of all the texture in the shaders.

Namespaces

- shader

7.116 src/shader/VAODataManager.cpp File Reference

```
#include "VAODataManager.hpp"
```

Functions

- template void VAODataManager::fill_bo< glm::vec2 > (GLenum buffer_type, GLuint vbo_id, std::vector< glm::vec2 > datas)
- template void VAODataManager::fill_bo< glm::vec3 > (GLenum buffer_type, GLuint vbo_id, std::vector< glm::vec3 > datas)

7.116.1 Function Documentation

7.116.1.1 VAODataManager::fill_bo< glm::vec2 >()

```
template void VAODataManager::fill_bo< glm::vec2 > (
            GLenum buffer_type,
            GLuint vbo_id,
            std::vector< glm::vec2 > datas )
```

7.116.1.2 VAODataManager::fill_bo< glm::vec3 >()

```
template void VAODataManager::fill_bo< glm::vec3 > (
            GLenum buffer_type,
            GLuint vbo_id,
            std::vector< glm::vec3 > datas )
```

7.117 src/shader/VAODataManager.hpp File Reference

```
#include <string>
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <iostream>
#include <vector>
```

Classes

- class shader::VAODataManager

    Manager of VBOs and EBO.

Namespaces

- shader

7.118 src/utils/Geometry3D.cpp File Reference

```
#include "Geometry3D.hpp"
```

7.119 src/utils/Geometry3D.hpp File Reference

```
#include <glm/glm.hpp>
```

Classes

- struct utils::Line

    Line with a start point and an end point.
- struct utils::Ray

    Ray with an origine and a direction.
- struct utils::Plane

    Plane with a normal and a distance from the origin along the axis of the normal.
- struct utils::Interval

    Represent an interval between 2 float values.

Namespaces

- physics
- physics::bounding_box
- utils

Macros

- #define cmp_float(x, y)

  Compare 2 floats.

7.119.1 Macro Definition Documentation

7.119.1.1 cmp_float

```
#define cmp_float(
               x,
               y )
```

Value:
```
    (fabsf((x)-(y)) <= FLT_EPSILON * \
        fmaxf(1.0f,                  \
        fmaxf(fabsf(x), fabsf(y)))   \
)
```

Compare 2 floats.

7.120 src/utils/meshloader.cpp File Reference

```
#include "meshloader.hpp"
#include <utility>
```

Functions

- MeshData create_plane (int nb_vertex_1, int nb_vertex_2, vec3 pos_vertex_start, vec3 pos_vertex_end, NormalDirection normal_dir)
- MeshData create_sphere (float radius, int slices, int stacks)

  Create a sphere Mesh.
- MeshData create_rectangle_cuboid (glm::vec3 size)

  Create a rectangle cuboid Mesh.

7.120.1 Function Documentation

7.120.1.1 create_plane()

```
MeshData create_plane (
             int nb_vertex_1,
             int nb_vertex_2,
             vec3 pos_vertex_start,
             vec3 pos_vertex_end,
             NormalDirection normal_dir )
```

7.120.1.2 create_rectangle_cuboid()

```
MeshData create_rectangle_cuboid (
             glm::vec3 size )
```

Create a rectangle cuboid Mesh.

Parameters

size	

Returns

cube mesh datas

7.120.1.3 create_sphere()

```
MeshData create_sphere (
             float radius = 0.5,
             int slices = 32,
             int stacks = 16 )
```

Create a sphere Mesh.

Parameters

radius	
slices	
stacks	

Returns

sphere mesh datas

7.121 src/utils/meshloader.hpp File Reference

```
#include <glm/glm.hpp>
```

```
#include <glm/gtc/matrix_transform.hpp>
#include <src/mesh/Mesh.hpp>
```

Enumerations

- enum NormalDirection {
  X_NORMAL_DIRECTION, Y_NORMAL_DIRECTION, Z_NORMAL_DIRECTION, X_INV_NORMAL_DIRECTION, Y_INV_NORMAL_DIRECTION, Z_INV_NORMAL_DIRECTION }

Functions

- MeshData create_plane (int nb_vertex_1, int nb_vertex_2, glm::vec3 pos_vertex_start, glm::vec3 pos_↩ vertex_end, NormalDirection normal_dir)

  Create a plane Mesh.
- MeshData create_sphere (float radius=0.5, int slices=32, int stacks=16)

  Create a sphere Mesh.
- MeshData create_rectangle_cuboid (glm::vec3 size)

  Create a rectangle cuboid Mesh.

7.121.1 Enumeration Type Documentation

7.121.1.1 NormalDirection

```
enum NormalDirection
```

Enumerator

X_NORMAL_DIRECTION	
Y_NORMAL_DIRECTION	
Z_NORMAL_DIRECTION	
X_INV_NORMAL_DIRECTION	
Y_INV_NORMAL_DIRECTION	
Z_INV_NORMAL_DIRECTION	

7.121.2 Function Documentation

7.121.2.1 create_plane()

```
MeshData create_plane (
            int nb_vertex_1,
            int nb_vertex_2,
            glm::vec3 pos_vertex_start,
            glm::vec3 pos_vertex_end,
            NormalDirection normal_dir )
```

Create a plane Mesh.

Parameters

nb_vertex_1	first resolution
nb_vertex_2	second resolution
pos_vertex_start	first vertex
pos_vertex_end	last vertex
normal	normal direction

Returns

plane mesh datas

7.121.2.2 create_rectangle_cuboid()

```
MeshData create_rectangle_cuboid (
              glm::vec3 size )
```

Create a rectangle cuboid Mesh.

Parameters

size	

Returns

cube mesh datas

7.121.2.3 create_sphere()

```
MeshData create_sphere (
              float radius = 0.5,
              int slices = 32,
              int stacks = 16 )
```

Create a sphere Mesh.

Parameters

radius	
slices	
stacks	

Returns

sphere mesh datas

7.122 src/utils/objloader.cpp File Reference

```
#include <vector>
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <cstring>
#include <fstream>
#include "glm/glm.hpp"
#include "objloader.hpp"
```

Functions

- bool loadOBJ (const char ∗path, std::vector< glm::vec3 > &out_vertices, std::vector< unsigned short int > &out_indices, std::vector< glm::vec2 > &out_uvs, std::vector< glm::vec3 > &out_normals)
- bool loadOFF (const std::string &filename, std::vector< glm::vec3 > &vertices, std::vector< unsigned short > &indices, std::vector< std::vector< unsigned short > > &triangles)
- bool loadOFF (const std::string &filename, std::vector< glm::vec3 > &vertices, std::vector< unsigned short > &faces)

7.122.1 Function Documentation

7.122.1.1 loadOBJ()

```
bool loadOBJ (
            const char * path,
            std::vector< glm::vec3 > & out_vertices,
            std::vector< unsigned short int > & out_indices,
            std::vector< glm::vec2 > & out_uvs,
            std::vector< glm::vec3 > & out_normals )
```

7.122.1.2 loadOFF() [1/2]

```
bool loadOFF (
            const std::string & filename,
            std::vector< glm::vec3 > & vertices,
            std::vector< unsigned short > & faces )
```

7.122.1.3 loadOFF() [2/2]

```
bool loadOFF (
            const std::string & filename,
            std::vector< glm::vec3 > & vertices,
            std::vector< unsigned short > & indices,
            std::vector< std::vector< unsigned short > > & triangles )
```

7.123 src/utils/objloader.hpp File Reference

```
#include <string>
#include <iostream>
#include <vector>
#include <algorithm>
```

Functions

- bool loadOBJ (const char *path, std::vector< glm::vec3 > &out_vertices, std::vector< unsigned short int > &out_indices, std::vector< glm::vec2 > &out_uvs, std::vector< glm::vec3 > &out_normals)
- bool loadAssImp (const char *path, std::vector< unsigned short > &indices, std::vector< glm::vec3 > &vertices, std::vector< glm::vec2 > &uvs, std::vector< glm::vec3 > &normals)
- bool loadOFF (const std::string &filename, std::vector< glm::vec3 > &vertices, std::vector< unsigned short > &faces)
- bool loadOFF (const std::string &filename, std::vector< glm::vec3 > &vertices, std::vector< unsigned short > &indices, std::vector< std::vector< unsigned short > > &triangles)

7.123.1 Function Documentation

7.123.1.1 loadAssImp()

```
bool loadAssImp (
            const char * path,
            std::vector< unsigned short > & indices,
            std::vector< glm::vec3 > & vertices,
            std::vector< glm::vec2 > & uvs,
            std::vector< glm::vec3 > & normals )
```

7.123.1.2 loadOBJ()

```
bool loadOBJ (
            const char * path,
            std::vector< glm::vec3 > & out_vertices,
            std::vector< unsigned short int > & out_indices,
            std::vector< glm::vec2 > & out_uvs,
            std::vector< glm::vec3 > & out_normals )
```

7.123.1.3 loadOFF() [1/2]

```
bool loadOFF (
            const std::string & filename,
            std::vector< glm::vec3 > & vertices,
            std::vector< unsigned short > & faces )
```

7.123.1.4 loadOFF() [2/2]

```
bool loadOFF (
            const std::string & filename,
            std::vector< glm::vec3 > & vertices,
            std::vector< unsigned short > & indices,
            std::vector< std::vector< unsigned short > > & triangles )
```

7.124 src/utils/printer.cpp File Reference

```
#include <src/utils/printer.hpp>
```

Functions

- void print_vec3 (glm::vec3 v)

  print a glm::vec3
- void print_vec4 (glm::vec4 v)

  print a glm::vec4
- void print_mat4 (glm::mat4 m)

  print a glm::mat4
- void print_mat3 (glm::mat3 m)

  print a glm::mat3

7.124.1 Function Documentation

7.124.1.1 print_mat3()

```
void print_mat3 (
            glm::mat3 m )
```

print a glm::mat3

Parameters

m	

7.124.1.2 print_mat4()

```
void print_mat4 (
            glm::mat4 m )
```

print a glm::mat4

Parameters

m	

7.124.1.3 print_vec3()

```
void print_vec3 (
            glm::vec3 v )
```

print a glm::vec3

Parameters

v	

7.124.1.4 print_vec4()

```
void print_vec4 (
            glm::vec4 v )
```

print a glm::vec4

Parameters

v	

7.125 src/utils/printer.hpp File Reference

```
#include <vector>
#include <iostream>
#include <glm/glm.hpp>
```

Functions

- void print_vec3 (glm::vec3 v)

  print a glm::vec3
- void print_vec4 (glm::vec4 v)

  print a glm::vec4
- void print_mat4 (glm::mat4 m)

  print a glm::mat4
- void print_mat3 (glm::mat3 m)

  print a glm::mat3

7.125.1 Function Documentation

7.125.1.1 print_mat3()

```
void print_mat3 (
             glm::mat3 m )
```

print a glm::mat3

Parameters

m	

7.125.1.2 print_mat4()

```
void print_mat4 (
             glm::mat4 m )
```

print a glm::mat4

Parameters

m	

7.125.1.3 print_vec3()

```
void print_vec3 (
             glm::vec3 v )
```

print a glm::vec3

Parameters

v	

7.125.1.4 print_vec4()

```
void print_vec4 (
            glm::vec4 v )
```

print a glm::vec4

Parameters

v	

7.126 src/utils/quaternion_utils.cpp File Reference

```
#include <glm/gtc/quaternion.hpp>
#include <glm/gtx/quaternion.hpp>
#include <glm/gtx/euler_angles.hpp>
#include <glm/gtx/norm.hpp>
#include "quaternion_utils.hpp"
```

Functions

- quat RotationBetweenVectors (vec3 start, vec3 dest)
- quat LookAt (vec3 direction, vec3 desiredUp)
- quat RotateTowards (quat q1, quat q2, float maxAngle)
- void tests ()

7.126.1 Function Documentation

7.126.1.1 LookAt()

```
quat LookAt (
            vec3 direction,
            vec3 desiredUp )
```

7.126.1.2 RotateTowards()

```
quat RotateTowards (
            quat q1,
            quat q2,
            float maxAngle )
```

7.126.1.3 RotationBetweenVectors()

```
quat RotationBetweenVectors (
            vec3 start,
            vec3 dest )
```

7.126.1.4 tests()

```
void tests ( )
```

7.127 src/utils/quaternion_utils.hpp File Reference

Functions

- quat RotationBetweenVectors (vec3 start, vec3 dest)
- quat LookAt (vec3 direction, vec3 desiredUp)
- quat RotateTowards (quat q1, quat q2, float maxAngle)

7.127.1 Function Documentation

7.127.1.1 LookAt()

```
quat LookAt (
            vec3 direction,
            vec3 desiredUp )
```

7.127.1.2 RotateTowards()

```
quat RotateTowards (
            quat q1,
            quat q2,
            float maxAngle )
```

7.127.1.3 RotationBetweenVectors()

```
quat RotationBetweenVectors (
            vec3 start,
            vec3 dest )
```

7.128 src/utils/tangentspace.cpp File Reference

```
#include <vector>
#include <glm/glm.hpp>
#include "tangentspace.hpp"
```

Functions

- void computeTangentBasis (std::vector< glm::vec3 > &vertices, std::vector< glm::vec2 > &uvs, std::vector< glm::vec3 > &normals, std::vector< glm::vec3 > &tangents, std::vector< glm::vec3 > &bitangents)

7.128.1 Function Documentation

7.128.1.1 computeTangentBasis()

```
void computeTangentBasis (
            std::vector< glm::vec3 > & vertices,
            std::vector< glm::vec2 > & uvs,
            std::vector< glm::vec3 > & normals,
            std::vector< glm::vec3 > & tangents,
            std::vector< glm::vec3 > & bitangents )
```

7.129 src/utils/tangentspace.hpp File Reference

Functions

- void computeTangentBasis (std::vector< glm::vec3 > &vertices, std::vector< glm::vec2 > &uvs, std::vector< glm::vec3 > &normals, std::vector< glm::vec3 > &tangents, std::vector< glm::vec3 > &bitangents)

7.129.1 Function Documentation

7.129.1.1 computeTangentBasis()

```
void computeTangentBasis (
            std::vector< glm::vec3 > & vertices,
            std::vector< glm::vec2 > & uvs,
            std::vector< glm::vec3 > & normals,
            std::vector< glm::vec3 > & tangents,
            std::vector< glm::vec3 > & bitangents )
```

7.130 src/utils/text2D.cpp File Reference

```
#include <vector>
#include <cstring>
#include <GL/glew.h>
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include "src/shader/shader.hpp"
#include "texture.hpp"
#include "text2D.hpp"
```

Functions

- void initText2D (const char ∗texturePath)
- void printText2D (const char ∗text, int x, int y, int size)
- void cleanupText2D ()

Variables

- unsigned int Text2DTextureID
- unsigned int Text2DVertexBufferID
- unsigned int Text2DUVBufferID
- unsigned int Text2DShaderID
- unsigned int Text2DUniformID

7.130.1 Function Documentation

7.130.1.1 cleanupText2D()

```
void cleanupText2D ( )
```

7.130.1.2 initText2D()

```
void initText2D (
            const char * texturePath )
```

7.130.1.3 printText2D()

```
void printText2D (
            const char * text,
            int x,
            int y,
            int size )
```

7.130.2 Variable Documentation

7.130.2.1 Text2DShaderID

```
unsigned int Text2DShaderID
```

7.130.2.2 Text2DTextureID

```
unsigned int Text2DTextureID
```

7.130.2.3 Text2DUniformID

```
unsigned int Text2DUniformID
```

7.130.2.4 Text2DUVBufferID

```
unsigned int Text2DUVBufferID
```

7.130.2.5 Text2DVertexBufferID

```
unsigned int Text2DVertexBufferID
```

7.131 src/utils/text2D.hpp File Reference

Functions

- void initText2D (const char ∗texturePath)
- void printText2D (const char ∗text, int x, int y, int size)
- void cleanupText2D ()

7.131.1 Function Documentation

7.131.1.1 cleanupText2D()

```
void cleanupText2D ( )
```

7.131.1.2 initText2D()

```
void initText2D (
             const char ∗ texturePath )
```

7.131.1.3 printText2D()

```
void printText2D (
             const char ∗ text,
             int x,
             int y,
             int size )
```

7.132 src/utils/texture.cpp File Reference

```
#include <src/utils/texture.hpp>
```

Macros

- #define FOURCC_DXT1 0x31545844
- #define FOURCC_DXT3 0x33545844
- #define FOURCC_DXT5 0x35545844

Functions

- GLuint load_bmp_custom (const std::string &path, int id_texture)
- GLuint loadDDS (const char ∗imagepath)

7.132.1 Macro Definition Documentation

7.132.1.1 FOURCC_DXT1

```
#define FOURCC_DXT1 0x31545844
```

7.132.1.2 FOURCC_DXT3

```
#define FOURCC_DXT3 0x33545844
```

7.132.1.3 FOURCC_DXT5

```
#define FOURCC_DXT5 0x35545844
```

7.132.2 Function Documentation

7.132.2.1 load_bmp_custom()

```
GLuint load_bmp_custom (
            const std::string & path,
            int id_texture )
```

7.132.2.2 loadDDS()

```
GLuint loadDDS (
            const char * imagepath )
```

7.133 src/utils/texture.hpp File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <string>
#include "GL/glew.h"
#include "GLFW/glfw3.h"
```

Functions

- GLuint load_bmp_custom (const std::string &path, int id_texture)
- GLuint loadDDS (const char ∗imagepath)

7.133.1 Function Documentation

7.133.1.1 load_bmp_custom()

```
GLuint load_bmp_custom (
            const std::string & path,
            int id_texture )
```

7.133.1.2 loadDDS()

```
GLuint loadDDS (
            const char * imagepath )
```

7.134 src/utils/Transform.cpp File Reference

```
#include "Transform.hpp"
#include <glm/gtc/type_ptr.hpp>
#include <cmath>
```

Functions

- bool operator== (const Transform &trsf1, const Transform &trsf2)
- bool operator!= (const Transform &trsf1, const Transform &trsf2)

7.134.1 Function Documentation

7.134.1.1 operator"!=()

```
bool operator!= (
            const Transform & trsf1,
            const Transform & trsf2 )
```

Parameters

trsf1	
trsf2	

Returns

is_inequal

7.134.1.2 operator==()

```
bool operator== (
            const Transform & trsf1,
            const Transform & trsf2 )
```

Parameters

trsf1	
trsf2	

Returns

is_equal

7.135 src/utils/Transform.hpp File Reference

```
#include <glm/glm.hpp>
#include <glm/gtc/matrix_transform.hpp>
#include <iostream>
#include <vector>
```

Classes

- struct TransformDirty

  State of a Transform.
- class Transform

  Class representing a 4 by 4 Matrix (Translation + Rotation + Scale)

Macros

- #define ORDER_ZYX 0
- #define ORDER_ZXY 1
- #define ORDER_YXZ 2
- #define ORDER_YZX 3
- #define ORDER_XYZ 4
- #define ORDER_XZY 5

7.135.1 Macro Definition Documentation

7.135.1.1 ORDER_XYZ

```
#define ORDER_XYZ 4
```

7.135.1.2 ORDER_XZY

```
#define ORDER_XZY 5
```

7.135.1.3 ORDER_YXZ

```
#define ORDER_YXZ 2
```

7.135.1.4 ORDER_YZX

```
#define ORDER_YZX 3
```

7.135.1.5 ORDER_ZXY

```
#define ORDER_ZXY 1
```

7.135.1.6 ORDER_ZYX

```
#define ORDER_ZYX 0
```

Index