Bomb Mario

Generated by Doxygen 1.8.17

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

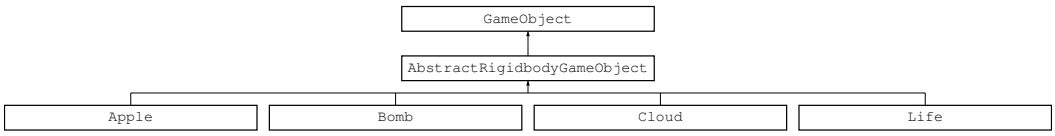
Class Documentation

4.1 AbstractRigidbodyGameObject Class Reference

AbstractRigidbodyGameObject class.

`#include <AbstractRigidBodyGameObject.hpp>`

Inheritance diagram for AbstractRigidbodyGameObject:



Public Member Functions

- AbstractRigidbodyGameObject ()
- AbstractRigidbodyGameObject (int width, int height)
- virtual void update (float deltaTime)=0
- virtual bool isRigidbody ()
- virtual void draw (sf::RenderWindow &window)=0
- virtual void setPosition (int x, int y)
- virtual sf::Vector2f getPosition ()
- virtual sf::RectangleShape getBody ()
- virtual bool hasCollidedWithPlayer ()
- virtual void setCollidedWithPlayer (bool collidedWithPlayer)
- virtual bool disposeObject ()

Protected Attributes

- bool _isRigidbody

  whether object is rigidbody
- bool _disposeObject

  object has to be removed form the memory
- bool _collidedWithPlayer

  object collided with player
- sf::RectangleShape _body

  object rectangular body

4.1.1 Detailed Description

AbstractRigidbodyGameObject class.

Moving game object in game Inherits GameObject class - every moving object also has a static object properties

4.1.2 Constructor & Destructor Documentation

4.1.2.1 AbstractRigidbodyGameObject() [1/2]

```
AbstractRigidbodyGameObject::AbstractRigidbodyGameObject ( )  [inline]
```

Basic class constructor without any parameters - empty object

4.1.2.2 AbstractRigidbodyGameObject() [2/2]

```
AbstractRigidbodyGameObject::AbstractRigidbodyGameObject (
            int width,
            int height )  [inline]
```

Class constructor with adjustable width and height of an moving object

Parameters

object	width
object	height

4.1.3 Member Function Documentation

4.1.3.1 disposeObject()

```
virtual bool AbstractRigidbodyGameObject::disposeObject ( )  [inline], [virtual]
```

Set object status to be removed form the memory

4.1.3.2 draw()

```
virtual void AbstractRigidbodyGameObject::draw (
            sf::RenderWindow & window )  [pure virtual]
```

Draw object on the screen This is pure virtual method so it must be overwritten in inheriting classes

Implemented in Bomb, Cloud, Apple, and Life.

4.1.3.3 getBody()

```
virtual sf::RectangleShape AbstractRigidbodyGameObject::getBody ( )  [inline], [virtual]
```

Get body of the player - rectangular box

Reimplemented in Bomb, Cloud, Apple, and Life.

4.1.3.4 getPosition()

```
virtual sf::Vector2f AbstractRigidbodyGameObject::getPosition ( )  [inline], [virtual]
```

Return position of object

4.1.3.5 hasColldedWithPlayer()

```
virtual bool AbstractRigidbodyGameObject::hasColldedWithPlayer ( )  [inline], [virtual]
```

Returns if moving object collided with player

4.1.3.6 isRigidbody()

```
virtual bool AbstractRigidbodyGameObject::isRigidbody ( )  [inline], [virtual]
```

Return whether given object is rigidbody

4.1.3.7 setColldedWithPlayer()

```
virtual void AbstractRigidbodyGameObject::setColldedWithPlayer (
            bool colldedWithPlayer )  [inline], [virtual]
```

Set object status to already collided with player

4.1.3.8 setPosition()

```
virtual void AbstractRigidbodyGameObject::setPosition (
            int x,
            int y )  [inline], [virtual]
```

Set position of object

Parameters

horizontal	positon
vertical	position

4.1.3.9 update()

```
virtual void AbstractRigidbodyGameObject::update (
            float deltaTime )  [pure virtual]
```

Update the logic of an moving object Because every moving object behaves differently this is pure virtual method so it must be overwritten in inheriting classes

Implemented in Bomb, Cloud, Apple, and Life.

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

- rigidbody_objects/AbstractRigidBodyGameObject.hpp

4.2 Animation Class Reference

Animation class.

```
#include <Animation.hpp>
```

Public Member Functions

- Animation (sf::Texture *texture, sf::Vector2u imageCount, float switchTime)
- Animation (sf::Texture *texture, sf::Vector2u imageCount, float switchTime, float currentImage)
- void update (int row, float deltaTime, bool flipX, bool secondRow, bool defaultV)

Public Attributes

- sf::IntRect uvrect

4.2.1 Detailed Description

Animation class.

Animates a game object

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Animation() [1/2]

```
Animation::Animation (
            sf::Texture * texture,
            sf::Vector2u imageCount,
            float switchTime )
```

Basic class constructor

Parameters

sprite	sheet texuture
number	of horizontal and vertical images
time	after which the picture change

4.2.2.2 Animation() [2/2]

```
Animation::Animation (
            sf::Texture * texture,
            sf::Vector2u imageCount,
            float switchTime,
            float currentImage )
```

Class constructor with the option of choosing the first texture

Parameters

sprite	sheet texuture
number	of horizontal and vertical images
time	after which the picture change
first	displayed image

4.2.3 Member Function Documentation

4.2.3.1 update()

```
void Animation::update (
            int row,
            float deltaTime,
            bool flipX,
            bool secondRow,
            bool defaultV )
```

Update sprie - change texture

Parameters

default	row
time	difference between previous frame
flip	horizontal texture
default	vertical image

4.2.4 Member Data Documentation

4.2.4.1 uvrect

`sf::IntRect Animation::uvrect`

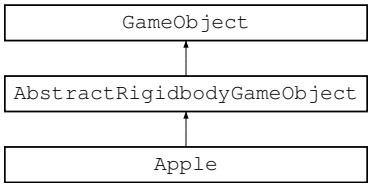
Box with current texture

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

- Animation.hpp
- Animation.cpp

4.3 Apple Class Reference

Inheritance diagram for Apple:



Public Member Functions

- **Apple** (sf::Texture *texture, sf::Vector2u dimensions, sf::Vector2u imageCount, sf::Vector2f bombSpeed, float bombAcceleration, float rotation)
- void update (float deltaTime) override
- void draw (sf::RenderWindow &window) override
- sf::RectangleShape getBody () override

Additional Inherited Members

4.3.1 Member Function Documentation

4.3.1.1 draw()

```
void Apple::draw (
            sf::RenderWindow & window )  [override], [virtual]
```

Draw object on the screen This is pure virtual method so it must be overwritten in inheriting classes

Implements AbstractRigidbodyGameObject.

4.3.1.2 getBody()

```
sf::RectangleShape Apple::getBody ( )  [override], [virtual]
```

Get body of the player - rectangular box

Reimplemented from AbstractRigidbodyGameObject.

4.3.1.3 update()

```
void Apple::update (
            float deltaTime )  [override], [virtual]
```

Update the logic of an moving object Because every moving object behaves differently this is pure virtual method so it must be overwritten in inheriting classes

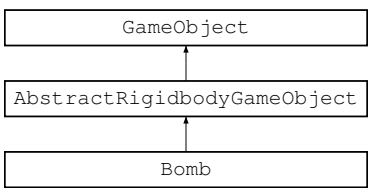
Implements AbstractRigidbodyGameObject.

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

- rigidbody_objects/Apple.hpp
- rigidbody_objects/Apple.cpp

4.4 Bomb Class Reference

Inheritance diagram for Bomb:



Public Member Functions

- **Bomb** (sf::Texture *texture, sf::Vector2u dimensions, sf::Vector2u imageCount, sf::Vector2f bombSpeed, float bombAcceleration, float rotation)
- void update (float deltaTime) override
- void draw (sf::RenderWindow &window) override
- sf::RectangleShape getBody () override
- void **setTexture** (sf::Texture *texture)

Additional Inherited Members

4.4.1 Member Function Documentation

4.4.1.1 draw()

```
void Bomb::draw (
            sf::RenderWindow & window )  [override], [virtual]
```

Draw object on the screen This is pure virtual method so it must be overwritten in inheriting classes

Implements AbstractRigidbodyGameObject.

4.4.1.2 getBody()

```
sf::RectangleShape Bomb::getBody ( )  [override], [virtual]
```

Get body of the player - rectangular box

Reimplemented from AbstractRigidbodyGameObject.

4.4.1.3 update()

```
void Bomb::update (
            float deltaTime )  [override], [virtual]
```

Update the logic of an moving object Because every moving object behaves differently this is pure virtual method so it must be overwritten in inheriting classes

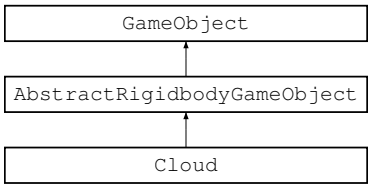
Implements AbstractRigidbodyGameObject.

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

- rigidbody_objects/Bomb.hpp
- rigidbody_objects/Bomb.cpp

4.5 Cloud Class Reference

Inheritance diagram for Cloud:



Public Member Functions

- **Cloud** (sf::Texture *texture, sf::Vector2u dimensions, sf::Vector2u imageCount, sf::Vector2f cloudSpeed, float shakeSpeed, float switchTime, float currentImage)
- void update (float deltaTime) override
- void draw (sf::RenderWindow &window) override
- sf::RectangleShape getBody () override

Additional Inherited Members

4.5.1 Member Function Documentation

4.5.1.1 draw()

```
void Cloud::draw (
            sf::RenderWindow & window )  [override], [virtual]
```

Draw object on the screen This is pure virtual method so it must be overwritten in inheriting classes

Implements AbstractRigidbodyGameObject.

4.5.1.2 getBody()

```
sf::RectangleShape Cloud::getBody ( )  [override], [virtual]
```

Get body of the player - rectangular box

Reimplemented from AbstractRigidbodyGameObject.

4.5.1.3 update()

```
void Cloud::update (
            float deltaTime )  [override], [virtual]
```

Update the logic of an moving object Because every moving object behaves differently this is pure virtual method so it must be overwritten in inheriting classes

Implements AbstractRigidbodyGameObject.

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

- rigidbody_objects/Cloud.hpp
- rigidbody_objects/Cloud.cpp

4.6 Game Class Reference

Game class.

```
#include <Game.hpp>
```

Classes

- struct Level

Public Types

- enum GameState { **RUNNING**, **EXITING**, **UNINITIALIZED**, **PAUSE** }

Public Member Functions

- void initGame ()
- void initRound ()
- void run (void)

Public Attributes

- float width
- float **height**
- std::string title

4.6.1 Detailed Description

Game class.

Controls all game objects, events and rendering

4.6.2 Member Enumeration Documentation

4.6.2.1 GameState

```
enum Game::GameState
```

All possible game states

4.6.3 Member Function Documentation

4.6.3.1 initGame()

```
void Game::initGame ( )
```

Initiation of a new game This method will re-initialize all dynamic content in the game

4.6.3.2 initRound()

```
void Game::initRound ( )
```

Initiation of a new round This method will initialize all variables in the game

4.6.3.3 run()

```
void Game::run (
            void  )
```

Method that is called to run the new game after the initialization of all variables

4.6.4 Member Data Documentation

4.6.4.1 title

```
std::string Game::title
```

Displayed window size

4.6.4.2 width

`float Game::width`

Displayed window size

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

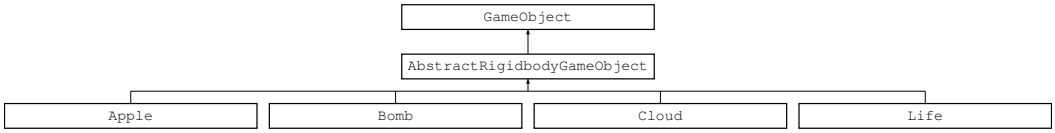
- Game.hpp
- Game.cpp

4.7 GameObject Class Reference

GameObject class.

`#include <GameObject.hpp>`

Inheritance diagram for GameObject:



Public Member Functions

- GameObject ()
- GameObject (int width, int height)
- ~GameObject ()
- void setPosition (float x, float y)
- sf::Vector2f getPosition ()
- void draw (sf::RenderWindow &window)
- void setTexture (std::string texturePath)
- sf::Texture getTexture ()
- bool setTextureRepeat (bool repeat)
- sf::IntRect getRectangle () const

Protected Attributes

- sf::IntRect _rect

  Rectangle of object.
- sf::Texture _texture

  Texture of object.
- sf::Sprite _sprite

  Sprite of object.
- bool _textureRepeat

  Texture repeat option.

4.7.1 Detailed Description

GameObject class.

Simple static object in game

4.7.2 Constructor & Destructor Documentation

4.7.2.1 GameObject() [1/2]

```
GameObject::GameObject ( )
```

Basic class constructor without any parameters

4.7.2.2 GameObject() [2/2]

```
GameObject::GameObject (
            int width,
            int height )
```

Class constructor with adjustable width and height of an object

Parameters

object	width
object	height

4.7.2.3 ~GameObject()

```
GameObject::~GameObject ( )
```

Class destructor

4.7.3 Member Function Documentation

4.7.3.1 draw()

```
void GameObject::draw (
            sf::RenderWindow & window )
```

Draw the object on selected window

4.7.3.2 getPosition()

```
sf::Vector2f GameObject::getPosition ( )
```

Return the size of object

4.7.3.3 getRectangle()

```
sf::IntRect GameObject::getRectangle ( ) const
```

Get rectangle of object

4.7.3.4 getTexture()

```
sf::Texture GameObject::getTexture ( )
```

Get texture of object

4.7.3.5 setPosition()

```
void GameObject::setPosition (
            float x,
            float y )
```

Adjusts the width and height of the object

4.7.3.6 setTexture()

```
void GameObject::setTexture (
            std::string texturePath )
```

Set the texture of object

4.7.3.7 setTextureRepeat()

```
bool GameObject::setTextureRepeat (
            bool repeat )
```

Set whatever the object's texture should be tiled

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

- GameObject.hpp
- GameObject.cpp

4.8 GameObjectManager< T > Class Template Reference

Game object manager.

`#include <GameObjectManager.hpp>`

Classes

- struct Node

  Node struct.

Public Member Functions

- GameObjectManager ()
- void operator+ (T const &obj)
- void operator- (T const &obj)
- void addItem (T *gameObject, std::string objectName)
- void deleteItem (Node *item)
- Node * findByObjectName (std::string objectName)
- Node * getNodeHead ()
- ~GameObjectManager ()

4.8.1 Detailed Description

template<typename T>
class GameObjectManager< T >

Game object manager.

Holds game objects Because it is a generic class all code is in header

4.8.2 Constructor & Destructor Documentation

4.8.2.1 GameObjectManager()

```
template<typename T >
GameObjectManager< T >::GameObjectManager ( )  [inline]
```

Basic class constructor null initialize node head and last item pointers

4.8.2.2 ~GameObjectManager()

```
template<typename T >
GameObjectManager< T >::~GameObjectManager ( )  [inline]
```

Destructor Deletes all items from the list.

4.8.3 Member Function Documentation

4.8.3.1 addItem()

```
template<typename T >
void GameObjectManager< T >::addItem (
            T * gameObject,
            std::string objectName )  [inline]
```

Adds a new item to the list

4.8.3.2 deleteItem()

```
template<typename T >
void GameObjectManager< T >::deleteItem (
            Node * item )  [inline]
```

Removes item form the list

4.8.3.3 findByObjectName()

```
template<typename T >
Node* GameObjectManager< T >::findByObjectName (
            std::string objectName )  [inline]
```

Finds a given object by the name

4.8.3.4 getNodeHead()

```
template<typename T >
Node* GameObjectManager< T >::getNodeHead ( )  [inline]
```

Get the head of list

4.8.3.5 operator+()

```
template<typename T >
void GameObjectManager< T >::operator+ (
            T const & obj )  [inline]
```

Operator overload Adds a new node to the list using "+" sign

4.8.3.6 operator-()

```
template<typename T >
void GameObjectManager< T >::operator- (
             T const & obj )  [inline]
```

Operator overload Removes node form the list using "-" sign

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

- GameObjectManager.hpp

4.9 Game::Level Struct Reference

```
#include <Game.hpp>
```

Public Attributes

- int levelDuration
  level duration
- std::string levelSlug
  level short name
- std::string levelDescription
  level description
- sf::Color levelColor
  color of level
- sf::Vector2f _bombDropFreqTimeRage
  frequency of bombs dropping
- sf::Vector2f _bombDropSpeedYRage
  speed of bomb dropping
- sf::Vector2f _appleDropFreqTimeRage
  frequency of apple dropping
- sf::Vector2f _cloudGenFreqTimeRage
  frequency of new clouds appearing
- sf::Vector2f _heartDropFreqTimeRage
  frequency of heart dropping

4.9.1 Detailed Description

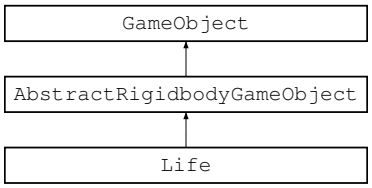
All information about levels in the game Such as frequency of bombs falling, bomb falling speed, frequency of new life falling etc.

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

- Game.hpp

4.10 Life Class Reference

Inheritance diagram for Life:



Public Member Functions

- **Life** (sf::Texture *texture, sf::Vector2u dimensions, sf::Vector2u imageCount, sf::Vector2f speed, float acceleration, float rotation)
- void update (float deltaTime) override
- void draw (sf::RenderWindow &window) override
- sf::RectangleShape getBody () override

Additional Inherited Members

4.10.1 Member Function Documentation

4.10.1.1 draw()

```
void Life::draw (
            sf::RenderWindow & window )  [override], [virtual]
```

Draw object on the screen This is pure virtual method so it must be overwritten in inheriting classes

Implements AbstractRigidbodyGameObject.

4.10.1.2 getBody()

```
sf::RectangleShape Life::getBody ( )  [override], [virtual]
```

Get body of the player - rectangular box

Reimplemented from AbstractRigidbodyGameObject.

4.10.1.3 update()

```
void Life::update (
            float deltaTime )  [override], [virtual]
```

Update the logic of an moving object Because every moving object behaves differently this is pure virtual method so it must be overwritten in inheriting classes

Implements AbstractRigidbodyGameObject.

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

- rigidbody_objects/Life.hpp
- rigidbody_objects/Life.cpp

4.11 GameObjectManager< T >::Node Struct Reference

Node struct.

```
#include <GameObjectManager.hpp>
```

Public Member Functions

- ~Node ()
- Node (T *gameObject, std::string objectName)
- Node & operator++ ()

Public Attributes

- T * gameObject

  Stored item.
- std::string objectName

  Stored item name.
- Node * next = nullptr

  Next element.
- Node * prev = nullptr

  Previous element.

4.11.1 Detailed Description

template<typename T>
struct GameObjectManager< T >::Node

Node struct.

One item in the list

4.11.2 Constructor & Destructor Documentation

4.11.2.1 ~Node()

```
template<typename T >
GameObjectManager< T >::Node::~Node ( )  [inline]
```

Removes stored item form memory

4.11.2.2 Node()

```
template<typename T >
GameObjectManager< T >::Node::Node (
            T * gameObject,
            std::string objectName )  [inline]
```

Constructor which creates a new node element

Parameters

object	to store
object	name

4.11.3 Member Function Documentation

4.11.3.1 operator++()

```
template<typename T >
Node& GameObjectManager< T >::Node::operator++ ( )  [inline]
```

Operator overload Select next element in the list using "++"

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

- GameObjectManager.hpp

4.12 Player Class Reference

```
#include <Player.hpp>
```

Public Member Functions

- Player (sf::Texture ∗texture, sf::Vector2u imageCount, float switchTime, float speed, float jumpHeight)
- ∼Player ()
- void update (float deltaTime)
- void draw (sf::RenderWindow &window)
- void setPosition (int x, int y)
- sf::Vector2f getPosition ()
- sf::RectangleShape getBody ()

4.12.1 Detailed Description

Defines player animations and behavior

4.12.2 Constructor & Destructor Documentation

4.12.2.1 Player()

```
Player::Player (
            sf::Texture * texture,
            sf::Vector2u imageCount,
            float switchTime,
            float speed,
            float jumpHeight )
```

Basic class constructor

Parameters

sprite	sheet of player texuture
number	of horizontal and vertical images
time	after which the player picture change
player	maximum speed
player	maximum jump height

4.12.2.2 ∼Player()

```
Player::∼Player ( )
```

Class destructor

4.12.3 Member Function Documentation

4.12.3.1 draw()

```
void Player::draw (
            sf::RenderWindow & window )
```

Draw player sprint on the screen

4.12.3.2 getBody()

```
sf::RectangleShape Player::getBody ( )
```

Get body of the player - rectangular box

4.12.3.3 getPosition()

```
sf::Vector2f Player::getPosition ( )
```

Return position of the player

4.12.3.4 setPosition()

```
void Player::setPosition (
            int x,
            int y )
```

Set position of the player

Parameters

horizontal	positon
vertical	position

4.12.3.5 update()

```
void Player::update (
            float deltaTime )
```

Update the logic of the player - movement and animation

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

- Player.hpp
- Player.cpp

4.13 Utils Class Reference

Utils.

```
#include <Utils.hpp>
```

Static Public Member Functions

- static float randomFloat (float min, float max)
- static int randomInt (int min, int max)
- static float calculateAngleOfRotationForVelocity (sf::Vector2f objectVelocity)
- static std::string resourcePath (void)

4.13.1 Detailed Description

Utils.

Useful tool used in many classes

4.13.2 Member Function Documentation

4.13.2.1 calculateAngleOfRotationForVelocity()

```
float Utils::calculateAngleOfRotationForVelocity (
            sf::Vector2f objectVelocity )  [static]
```

Calculate angle of rotation for velocity to tilt the object in the direction of falling -objectVelocity.x / objectVelocity.y)) ∗ 180) / PI = tan(x) where x is searched number

4.13.2.2 randomFloat()

```
float Utils::randomFloat (
            float min,
            float max )  [static]
```

Generate random float number

4.13.2.3 randomInt()

```
int Utils::randomInt (
            int min,
            int max )  [static]
```

Generate random integer number

4.13.2.4 resourcePath()

```
std::string Utils::resourcePath (
            void  )  [static]
```

Returns default assets folder

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

- Utils.hpp
- Utils.cpp

Chapter 5

File Documentation

5.1 Animation.hpp File Reference

```
#include <SFML/Graphics.hpp>
```

Classes

- class Animation

  Animation class.

5.2 Game.hpp File Reference

```
#include <SFML/Graphics.hpp>
#include "GameObject.hpp"
#include "GameObjectManager.hpp"
#include "Player.hpp"
#include "Utils.hpp"
#include "rigidbody_objects/AbstractRigidBodyGameObject.hpp"
#include "rigidbody_objects/Bomb.hpp"
#include "rigidbody_objects/Apple.hpp"
#include "rigidbody_objects/Cloud.hpp"
#include "rigidbody_objects/Life.hpp"
```

Classes

- class Game

  Game class.
- struct Game::Level

5.3 GameObject.hpp File Reference

```
#include <SFML/Graphics.hpp>
```

Classes

- class GameObject
  GameObject class.

5.4 GameObjectManager.hpp File Reference

```
#include <string>
```

Classes

- class GameObjectManager< T >
  Game object manager.
- struct GameObjectManager< T >::Node
  Node struct.

5.5 Player.hpp File Reference

```
#include "Animation.hpp"
```

Classes

- class Player

5.6 rigidbody_objects/AbstractRigidBodyGameObject.hpp File Reference

```
#include "../GameObject.hpp"
```

Classes

- class AbstractRigidbodyGameObject
  AbstractRigidbodyGameObject class.

5.7 rigidbody_objects/Apple.hpp File Reference

```
#include "AbstractRigidbodyGameObject.hpp"
#include "../Animation.hpp"
```

Classes

- class Apple

5.8 rigidbody_objects/Bomb.hpp File Reference

```
#include "AbstractRigidbodyGameObject.hpp"
#include "../GameObject.hpp"
#include "../Animation.hpp"
```

Classes

- class Bomb

5.9 rigidbody_objects/Cloud.hpp File Reference

```
#include "AbstractRigidbodyGameObject.hpp"
#include "../Animation.hpp"
#include "../Utils.hpp"
```

Classes

- class Cloud

5.10 rigidbody_objects/Life.hpp File Reference

```
#include "AbstractRigidbodyGameObject.hpp"
#include "../Animation.hpp"
```

Classes

- class Life

5.11 Utils.hpp File Reference

```
#include <cmath>
#include <cstdlib>
#include <SFML/System/Vector2.hpp>
#include <string>
```

Classes

- class Utils

  Utils.

Variables

- constexpr auto **PI** = 3.14159265