Computer Programing: Project

Generated by Doxygen 1.8.17

Chapter 1

README

Place your project here

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 documented files with brief descriptions:

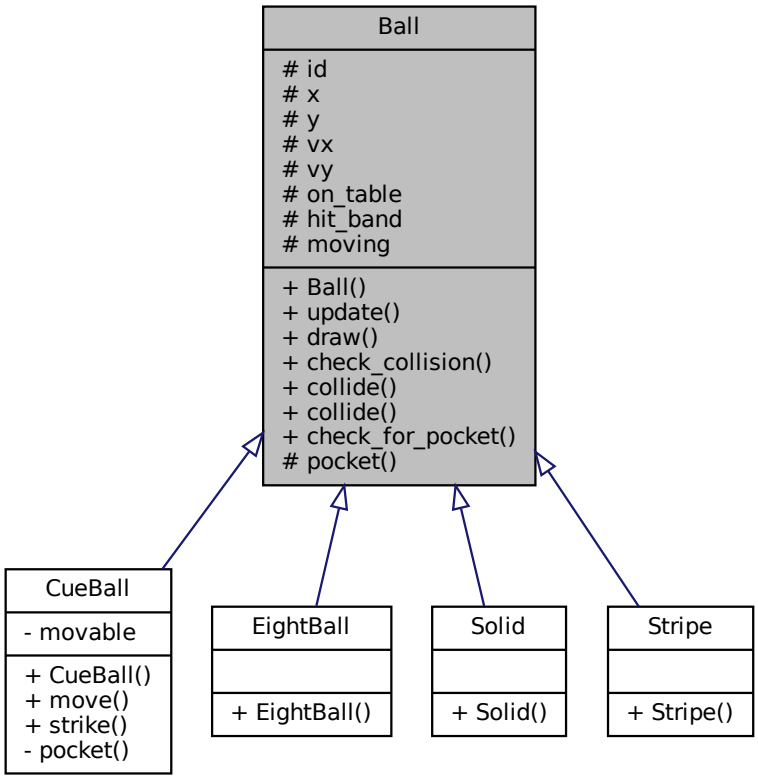
Chapter 5

Class Documentation

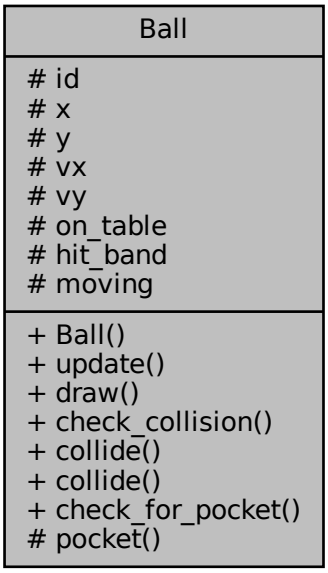
5.1 Ball Class Reference

`#include <balls.h>`

Inheritance diagram for Ball:



Collaboration diagram for Ball:



Public Member Functions

- Ball (double x, double y, int id)
- void update (Uint32 dt)
- void draw (Game ∗r)
- bool check_collision (std::shared_ptr< Ball > b)
- void collide (Segment ∗s)
- void collide (std::shared_ptr< Ball > b)
- bool check_for_pocket ()

Protected Member Functions

- virtual void pocket ()

  action taken after pocketing a ball, different for "normal" balls, cue ball and for eight ball

Protected Attributes

- int id

  Billard balls are numbered, cue ball is assumed to have id 0.
- double x

  x coordinate
- double y

  y coordinate
- double vx

x component of velocity

- double vy

  y component of velocity

- bool on_table
- bool hit_band
- bool moving

  flag set to true if ball is moving - used to determine end of turn

Friends

- class **Segment**
- class **Game**

5.1.1 Detailed Description

Class representing basic billard ball

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Ball()

```
Ball::Ball (
            double x,
            double y,
            int id )
```

ball requires specifying position and balls number

Parameters

x	x coordinate
y	y coordinate
id	number of the ball

5.1.3 Member Function Documentation

5.1.3.1 check_collision()

```
bool Ball::check_collision (
            std::shared_ptr< Ball > b )
```

function responsible for checking if collision with ball b occured

Parameters

b	shared pointer to ball to check collisions against

Returns

true if the collision occured

5.1.3.2 check_for_pocket()

```
bool Ball::check_for_pocket ( )
```

function checking if ball has been pocketed

Returns

function returns true if the ball has been pocketed

5.1.3.3 collide() [1/2]

```
void Ball::collide (
            Segment * s )
```

function responsible for handling collision with segment

Parameters

s	pointer to segment to collide with

5.1.3.4 collide() [2/2]

```
void Ball::collide (
            std::shared_ptr< Ball > b )
```

function responsible for handling collision with other ball

Parameters

b	shared pointer to ball to collide with

5.1.3.5 draw()

```
void Ball::draw (
            Game * r )
```

function responsible for displaying the ball

Parameters

r	pointer to Game object holding apropriate rendering context

5.1.3.6 update()

```
void Ball::update (
            Uint32 dt )
```

function responsible for physics and checking for potting and fauls

Parameters

dt	time elapsed from last update in milliseconds

5.1.4 Member Data Documentation

5.1.4.1 hit_band

```
bool Ball::hit_band  [protected]
```

this flag is set to true if ball hit the band in this turn to check if the faul has been commited

5.1.4.2 on_table

```
bool Ball::on_table  [protected]
```

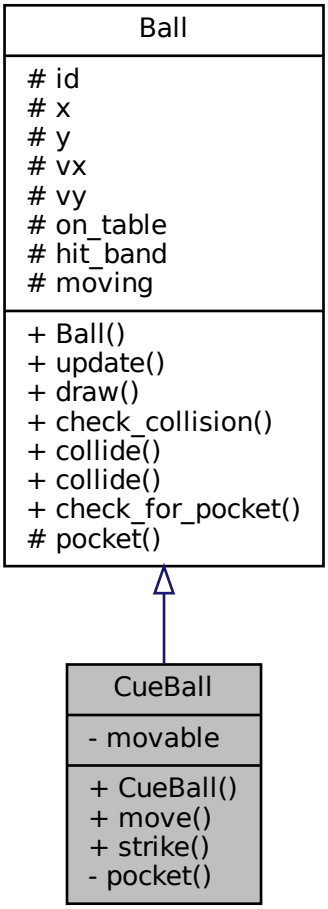
this flag is true if ball is still in the game, if it's false the ball isn't drawn neither checked for collisions

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

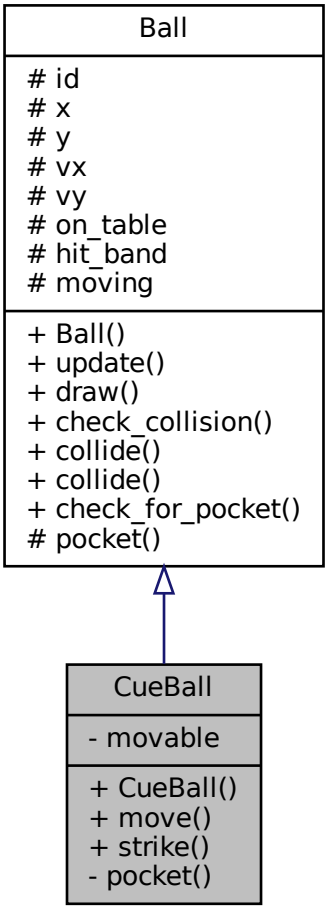
- balls.h
- balls.cpp

5.2 CueBall Class Reference

Inheritance diagram for CueBall:



Collaboration diagram for CueBall:



Public Member Functions

- CueBall (double x, double y)
- void move (double x, double y)
- void strike (double vx, double vy)

Private Member Functions

- void pocket ()

  cue ball after pocketing returns to the table

Private Attributes

- bool movable

  flag set if player has the ball in hand

Friends

- class **Game**

Additional Inherited Members

5.2.1 Constructor & Destructor Documentation

5.2.1.1 CueBall()

```
CueBall::CueBall (
            double x,
            double y )
```

cue ball always has id 0, so it doesn't need to be passed to constructor

Parameters

x	x coordinate
y	y coordinate

5.2.2 Member Function Documentation

5.2.2.1 move()

```
void CueBall::move (
            double x,
            double y )
```

function responsible for moving the ball if player has it in hand

Parameters

x	destination x coordinate
y	destination y corrdinate

5.2.2.2 strike()

```
void CueBall::strike (
```

```
            double vx,
            double vy )
```

function initiating movement of white ball

Parameters

vx	horizontal element of velocity
vy	vertical element of velocity

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

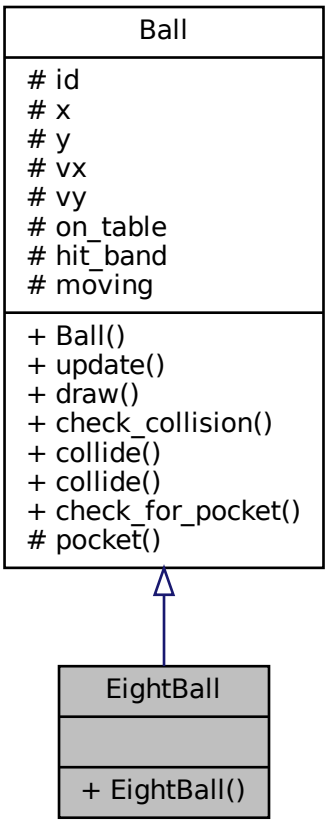
- balls.h
- balls.cpp

5.3 EightBall Class Reference

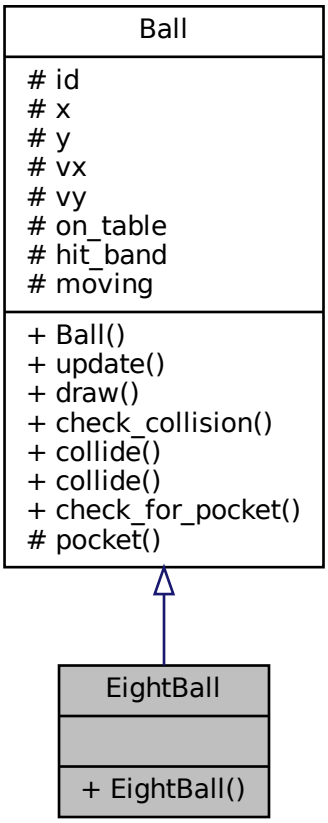
separate class for RTTI and to get constant id

`#include <balls.h>`

Inheritance diagram for EightBall:



Collaboration diagram for EightBall:



Public Member Functions

- EightBall (double x, double y)

Additional Inherited Members

5.3.1 Detailed Description

separate class for RTTI and to get constant id

5.3.2 Constructor & Destructor Documentation

5.3.2.1 EightBall()

```
EightBall::EightBall (
            double x,
            double y )
```

cue ball always has id 8, so it doesn't need to be passed to constructor

Parameters

x	x coordinate
y	y coordinate

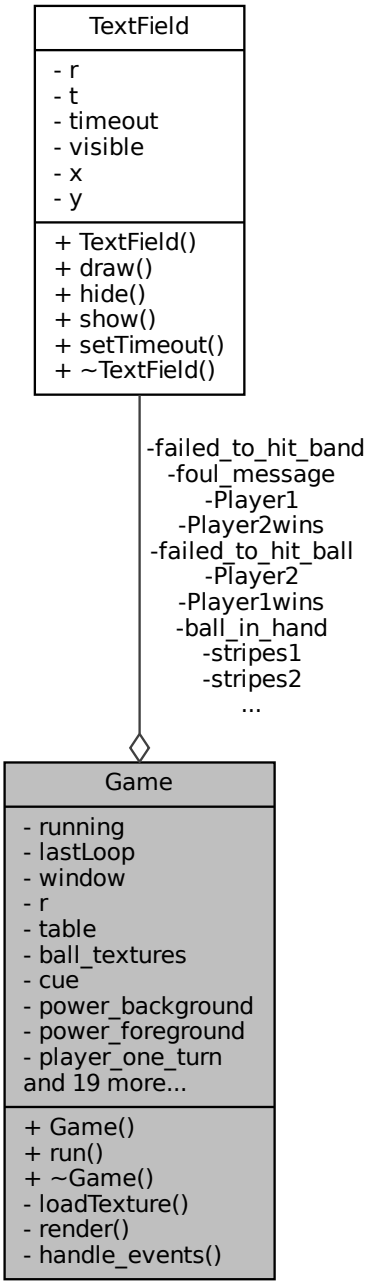
The documentation for this class was generated from the following files:

- balls.h

- balls.cpp

5.4 Game Class Reference

Collaboration diagram for Game:



Public Member Functions

- Game ()

  constructor taking care of initialising SDL, creating window and loading textures

- void run ()

  main loop of the game
- ~Game ()

  destructor takes care of proper release of SDL resources

Private Types

- enum {
  WAITING, **MOVING_BALL**, **MOVING_CUE**, **SETTING_POWER**,
  ROLLING, **END** }

  current state of the game

Private Member Functions

- SDL_Texture * loadTexture (const char *fname)
- void render ()

  function rendering elements that cannot be timeouted
- void handle_events ()

  function taking care of SDL events - moving mouse, keypresses, closing window e.t.c.

Private Attributes

- bool running

  if this flag is 0, the game stops
- Uint32 lastLoop

  variable for measuring time between frames
- SDL_Window * window

  pointer to the window
- SDL_Renderer * r

  pointer to SDL rendering context
- SDL_Texture * table

  pointer to table texture
- std::vector< SDL_Texture * > ball_textures

  vector of pointers to ball textures
- SDL_Texture * cue

  pointer to cue texture
- SDL_Texture * power_background

  pointer to texture of powerbar background
- SDL_Texture * power_foreground

  pointer to texture of powerbar foreground
- bool player_one_turn = true

  flag showing whose move it is
- std::vector< std::shared_ptr< Ball > > balls

  vector storing pointers to all balls(ones on the table and pocketed ones)
- enum Game:: { ... } state

  current state of the game
- int last_y

  variable tracking mouse movement along vertical axis to set strike power
- int solids_left = 7

number of solid balls left on table

- int stripes_left = 7

  number of striped balls left on table
- std::vector< Segment * > bands

  representation of boundaries of table
- double alpha =0

  angle of cues rotation
- double power = 0

  power of strike
- bool first_hit = true

  flag set if white ball didn't hit any other ball in current turn
- std::shared_ptr< Ball > player_one_balls

  pointer used for RTTI idenfification of balls belonging to player 1
- bool balls_assigned = false

  flag set to true if balls have been assigned, used for faul checking and sanity checks to not try typeid on nullptr
- bool break_shot = true

  flag set if it is first shot of the game
- bool right_ball_pocketed = false

  flag set if player pocketed at least one of his balls in current turn
- bool black_out_of_table = false

  flag set if black ball has been pocketed, pocketing it after break shot means end of the game
- bool foul = false

  flag set if foul has been commited in current turn
- bool ball_pocketed

  flag set if any ball has been pocketed in current turn - exception to foul when no ball hits band
- TextField * **Player1**
- TextField * **Player2**
- TextField * **ball_in_hand**
- TextField * **failed_to_hit_ball**
- TextField * **failed_to_hit_band**
- TextField * **foul_message**
- TextField * **Player1wins**
- TextField * **Player2wins**
- TextField * **pocketed_cue_ball**
- TextField * **solids1**
- TextField * **solids2**
- TextField * **stripes1**
- TextField * stripes2

  pointers to text fields that can be visible on screen

Friends

- class **Ball**

5.4.1 Member Function Documentation

5.4.1.1 loadTexture()

```
SDL_Texture * Game::loadTexture (
            const char * fname )  [private]
```

function loading textures from files

Parameters

fname	name of the graphics file to be loaded as texture

Returns

pointer to SDL texture

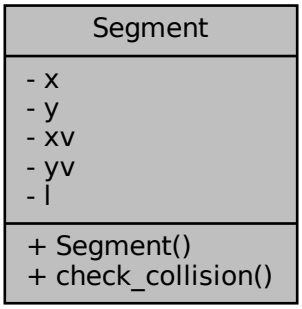
The documentation for this class was generated from the following files:

- classes.h
- classes.cpp

5.5 Segment Class Reference

`#include <classes.h>`

Collaboration diagram for Segment:



Public Member Functions

- Segment (double ax, double ay, double bx, double by)
- bool check_collision (std::shared_ptr< Ball > ball)

Private Attributes

- double x

  x coordinate of initial point
- double y

  y coordinate of initial point
- double xv

  x component of direction vector of the segment
- double yv

  y component of direction vector of the segment
- double l

  length of segment

Friends

- class **Ball**

5.5.1 Detailed Description

Class used to represent boundaries of the table and check for collisions

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Segment()

```
Segment::Segment (
            double ax,
            double ay,
            double bx,
            double by )
```

constructor generating segment from end points

Parameters

ax	x coordinate of initial point
ay	y coordinate of initial point
bx	x coordinate of terminal point
by	y coordinate of terminal point

5.5.3 Member Function Documentation

5.5.3.1 check_collision()

```
bool Segment::check_collision (
            std::shared_ptr< Ball > ball )
```

function for checking collision with ball

Parameters

ball	shared pointer to ball to check collision against

Returns

true if collision has occured

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

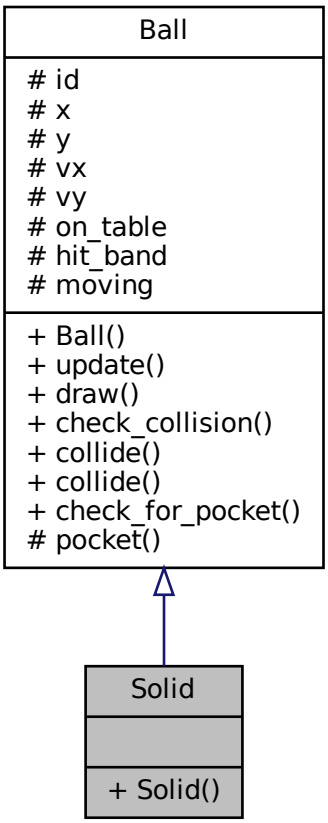
- classes.h
- classes.cpp

5.6 Solid Class Reference

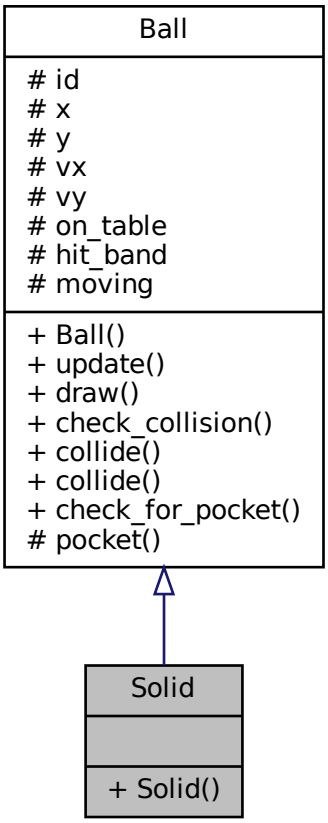
separate class for RTTI

```
#include <balls.h>
```

Inheritance diagram for Solid:



Collaboration diagram for Solid:



Public Member Functions

- Solid (double x, double y, int id)

Additional Inherited Members

5.6.1 Detailed Description

separate class for RTTI

5.6.2 Constructor & Destructor Documentation

5.6.2.1 Solid()

```
Solid::Solid (
            double x,
            double y,
            int id )
```

Parameters

x	x coordinate
y	y coordinate

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

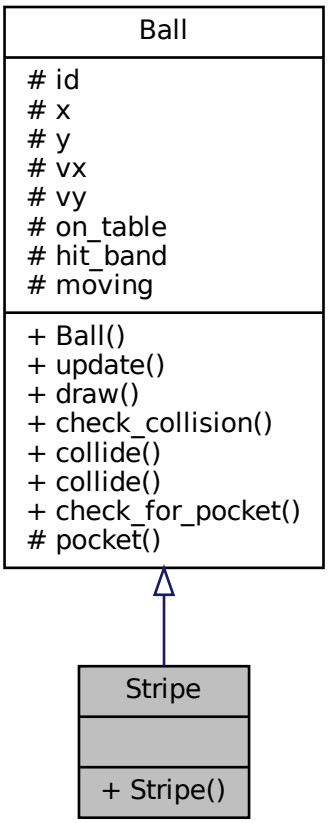
- balls.h
- balls.cpp

5.7 Stripe Class Reference

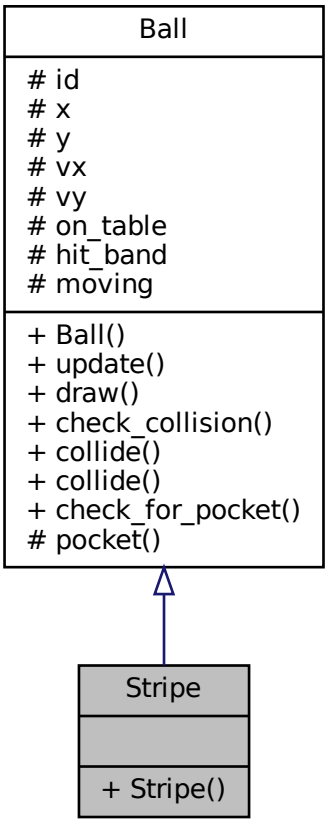
separate class for RTTI

`#include <balls.h>`

Inheritance diagram for Stripe:



Collaboration diagram for Stripe:



Public Member Functions

- Stripe (double x, double y, int id)

Additional Inherited Members

5.7.1 Detailed Description

separate class for RTTI

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Stripe()

```
Stripe::Stripe (
            double x,
            double y,
            int id )
```

Parameters

x	x coordinate
y	y coordinate

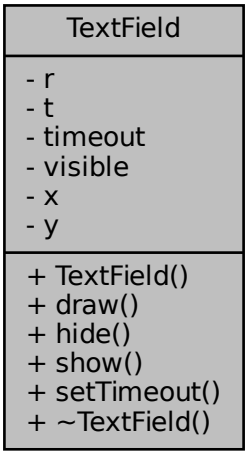
The documentation for this class was generated from the following files:

- balls.h
- balls.cpp

5.8 TextField Class Reference

`#include <classes.h>`

Collaboration diagram for TextField:



Public Member Functions

- TextField (SDL_Renderer *r, const char *text, TTF_Font *font, int x, int y)
- void draw (Uint32 dt)
- void hide ()

  function hiding text field
- void show ()

  function showing text field
- void setTimeout (int timeout)
- ~TextField ()

  destructor needed to destroy texture

Private Attributes

- SDL_Renderer * r

  pointer to SDL rendering context
- SDL_Texture * t

  pointer to texture containing text
- int timeout

  time left to vanish, -1 if text shouldn't be timed out
- bool visible

  flag set if text should be visible
- int x

  x coordinate
- int y

  y coordinate

5.8.1 Detailed Description

Class responsible for rendering text on screen

5.8.2 Constructor & Destructor Documentation

5.8.2.1 TextField()

```
TextField::TextField (
            SDL_Renderer * r,
            const char * text,
            TTF_Font * font,
            int x,
            int y )
```

constructor assigning values to the text field and generating the texture from text

Parameters

r	pointer to SDL rendering context
text	constant text to be rendered by this field
font	pointer to font
x	x coordinate
y	y coordinate

5.8.3 Member Function Documentation

5.8.3.1 draw()

```
void TextField::draw (
            Uint32 dt )
```

function responsible for drawing visible text field and updating time left on screen

Parameters

dt	time elapsed since last call to this function

5.8.3.2 setTimeout()

```
void TextField::setTimeout (
            int timeout )
```

function setting text field's timeout

Parameters

timeout	time for which the text field will be visible on screen, -1 for infinite

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

- classes.h
- classes.cpp

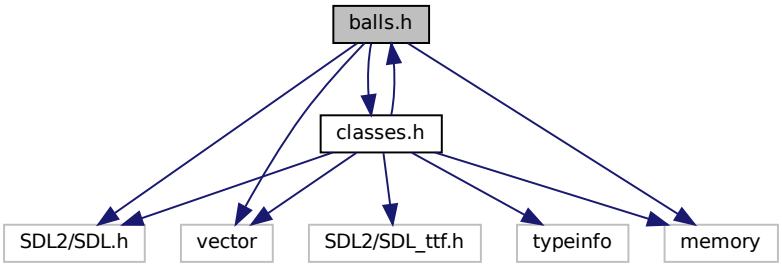
Chapter 6

File Documentation

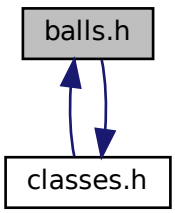
6.1 balls.h File Reference

```
#include <SDL2/SDL.h>
#include <vector>
#include <memory>
#include "classes.h"
```

Include dependency graph for balls.h:



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



Classes

- class Ball
- class CueBall
- class EightBall

  separate class for RTTI and to get constant id
- class Solid

  separate class for RTTI
- class Stripe

  separate class for RTTI

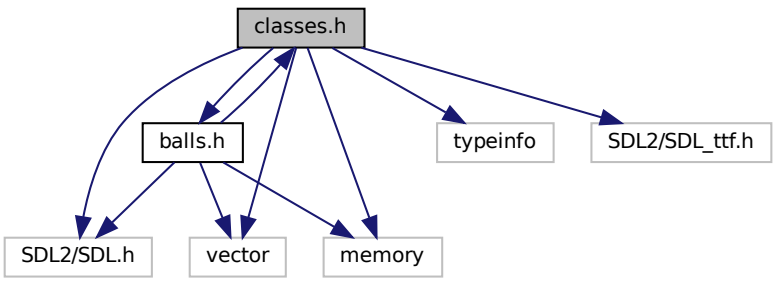
Macros

- #define **XOFF** 120
- #define **YOFF** 101
- #define **WIDTH** 924
- #define **HEIGHT** 461

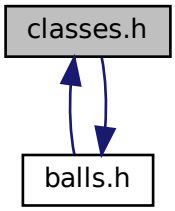
6.2 classes.h File Reference

```
#include "balls.h"
#include <SDL2/SDL.h>
#include <vector>
#include <memory>
#include <typeinfo>
#include <SDL2/SDL_ttf.h>
```

Include dependency graph for classes.h:



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



Classes

- class Segment
- class TextField
- class Game

Index