Bombás játék

Generated by Doxygen 1.8.5

Wed Oct 30 2013 01:15:45

Contents

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

Class Documentation

3.1 Animation Class Reference

Public Member Functions

- Animation ()
- Animation (Rect irect, TIME istart, AnimFile ∗ianim, std::string iparam="")
- std::string draw (TIME time)
- Rect getRect ()

3.1.1 Constructor & Destructor Documentation

3.1.1.1 Animation::Animation ( ) [inline]

Default constructor

3.1.1.2 Animation::Animation (Rect *irect*, TIME *istart*, AnimFile ∗ *ianim*, std::string *iparam* = " ") [inline]

Constructor

Parameters

irect	rectangle where the animation will be shown
istart	start time of animation
ianim	previously loaded animation file
iparam	drawing parameters

3.1.2 Member Function Documentation

3.1.2.1 std::string Animation::draw (TIME *time*) [inline]

Draw animation frame regarding time

Parameters

time	current frame time

Returns

graphic string

3.1.2.2 Rect Animation::getRect ( ) [inline]

Rectangle where animation is drawn

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

- view/animation.h

3.2 AnimFile Class Reference

Public Member Functions

- AnimFile ()
- AnimFile (std::string filename)
- std::string frame (TIME time)

3.2.1 Constructor & Destructor Documentation

3.2.1.1 AnimFile::AnimFile ( ) [inline]

Default constructor

3.2.1.2 AnimFile::AnimFile (std::string *filename*)

Constructor, loads animation phases

Parameters

filename	name of animation file (.anm)

3.2.2 Member Function Documentation

3.2.2.1 std::string AnimFile::frame (TIME *time*)

Calculate the current animation phase to show depending on current frame time

Parameters

time	frame time

Returns

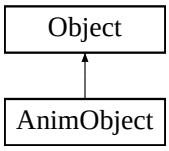
graphic name (full draw path)

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

- view/animfile.h
- view/animfile.cpp

3.3 AnimObject Class Reference

Inheritance diagram for AnimObject:



Public Member Functions

- AnimObject ()
- AnimObject (std::deque< Animation > *ianim, std::map< std::string, AnimFile > *ianims, std::string iname, TIME *inow)
- void draw (float x, float y, float w, float h, std::string param="")

3.3.1 Constructor & Destructor Documentation

3.3.1.1 AnimObject::AnimObject ( ) [inline]

Default constructor

3.3.1.2 AnimObject::AnimObject (std::deque< Animation > * *ianim*, std::map< std::string, AnimFile > * *ianims*, std::string *iname*, TIME * *inow*) [inline]

Constructor

Parameters

ianim	deque of animations from view
ianims	animation files
iname	name of animation
inow	pointer to current frame time

3.3.2 Member Function Documentation

3.3.2.1 void AnimObject::draw (float *x*, float *y*, float *w*, float *h*, std::string *param* = " ") [inline], [virtual]

Draw animation

Parameters

x	coordinate
y	coordinate
w	width
h	height
param	graphic parameter

Reimplemented from Object.

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

- view/animobject.h

3.4 Bomb Class Reference

Public Member Functions

- Bomb ()
- Bomb (TIME iactivated, Point< int > ipoint, Player *iplayer)
- void boom ()
- bool boom (TIME inow)
- std::string state ()
- int getX ()
- int getY ()
- Point< int > getPoint ()
- int getSize ()
- int getOwner ()

3.4.1 Constructor & Destructor Documentation

3.4.1.1 Bomb::Bomb ( ) [inline]

Default constructor

3.4.1.2 Bomb::Bomb (TIME *iactivated*, Point< int > *ipoint*, Player * *iplayer*) [inline]

Constructor

Parameters

iactivated	when should it become ignited
ipoint	place of bomb
iplayer	Player who put it

3.4.2 Member Function Documentation

3.4.2.1 void Bomb::boom ( ) [inline]

Bomb ignites by another flame, update player's number of bombs

3.4.2.2 bool Bomb::boom (TIME *inow*) [inline]

Bomb ignites by time

Parameters

inow	current frame time

Returns

ignites or not

3.4.2.3 int Bomb::getOwner ( ) [inline]

Get owner

Returns

player ID

3.4.2.4 Point<int> Bomb::getPoint ( ) [inline]

Get place of bomb

3.4.2.5 int Bomb::getSize ( ) [inline]

Get flame size

3.4.2.6 int Bomb::getX ( ) [inline]

Get X coordinate

3.4.2.7 int Bomb::getY ( ) [inline]

Get Y coordinate

3.4.2.8 std::string Bomb::state ( ) [inline]

Animation state

Returns

graphic string

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

- bomb.h

3.5 Client Class Reference

Public Member Functions

- void **mainLoop** ()

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

- client.h
- client.cpp

3.6 Config Struct Reference

Public Attributes

- int CFG_FLAMETIME
- int CFG_BOMBTIME
- float CFG_DEFAULT_SPEED
- float CFG_SLOW_SPEED
- float CFG_FAST_SPEED
- int CFG_VIRUSTIME
- int CFG_RESPAWN_PLAYER
- int CFG_RESPAWN_MONSTER
- int CFG_SHOW_SCOREBOARD
- unsigned int CFG_PORT

3.6.1 Member Data Documentation

3.6.1.1 int Config::CFG_BOMBTIME

Bomb ticks for that time

3.6.1.2 float Config::CFG_DEFAULT_SPEED

Default speed factor

3.6.1.3 float Config::CFG_FAST_SPEED

Fast virus speed factor

3.6.1.4 int Config::CFG_FLAMETIME

Flame lasts for that time after ignited

3.6.1.5 unsigned int Config::CFG_PORT

Port for multiplayer

3.6.1.6 int Config::CFG_RESPAWN_MONSTER

Monster spawning time interval

3.6.1.7 int Config::CFG_RESPAWN_PLAYER

Player respawn time

3.6.1.8 int Config::CFG_SHOW_SCOREBOARD

Show scoreboard for given time

3.6.1.9 float Config::CFG_SLOW_SPEED

Slow virus speed factor

3.6.1.10 int Config::CFG_VIRUSTIME

Virus time

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

- config.h
- config.cpp

3.7 Control Class Reference

Public Member Functions

- Control ()
- float px (float ix, float speed)
- float py (float iy, float speed)
- void setX (float x)
- void setY (float y)
- void setDelay (float d)
- void setPut (bool p)
- float getX ()
- float getY ()
- float getDelay ()
- bool getPut ()

3.7.1 Constructor & Destructor Documentation

3.7.1.1 Control::Control ( ) [inline]

Constructor

3.7.2 Member Function Documentation

3.7.2.1 float Control::getDelay ( ) [inline]

Get time difference between frames

3.7.2.2 bool Control::getPut ( ) [inline]

Get if the player wants to put bomb

3.7.2.3 float Control::getX ( ) [inline]

Get X vector

3.7.2.4 float Control::getY ( ) [inline]

Get Y vector

3.7.2.5 float Control::px (float *ix*, float *speed*) [inline]

Get new X coordinate regarding current position, speed and frame time

Parameters

ix	current X position
speed	speed factor

Returns

new X position

3.7.2.6 float Control::py (float *iy,* float *speed*) [inline]

Get new Y coordinate regarding current position, speed and frame time

Parameters

iy	current Y position
speed	speed factor

Returns

new Y position

3.7.2.7 void Control::setDelay (float *d*) [inline]

Set time difference between frames

3.7.2.8 void Control::setPut (bool *p*) [inline]

Set if the player wants to put bomb

3.7.2.9 void Control::setX (float *x*) [inline]

Set X vector

3.7.2.10 void Control::setY (float *y*) [inline]

Set Y vector

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

- control.h

3.8 Flame Class Reference

Public Member Functions

- Flame ()
- Flame (int x, int y, TIME itime, int idirection, int iOwner)
- int getX ()
- int getY ()
- Point< int > getPoint ()
- int getDirection ()
- int getOwner ()
- std::string state ()
- bool expire (TIME now)

Friends

- bool operator== (Flame &a, Point< int > &b)

3.8.1 Constructor & Destructor Documentation

3.8.1.1 Flame::Flame ( ) [inline]

Default constructor

3.8.1.2 Flame::Flame (int *x,* int *y,* TIME *itime,* int *idirection,* int *iOwner*)

Constructor, initialize values

Parameters

x	coordinate
y	coordinate
itime	flame expire time
idirection	direction, see getDirection()
iOwner	player ID

3.8.2 Member Function Documentation

3.8.2.1 bool Flame::expire (TIME *now*)

Whether flame is expired or not

Parameters

now	current frame time

3.8.2.2 int Flame::getDirection ( )

Get direction defined.

Returns

direction

3.8.2.3 int Flame::getOwner ( )

Return owner of flame

3.8.2.4 Point< int > Flame::getPoint ( )

Get point

3.8.2.5 int Flame::getX ( )

Get X coordinate

3.8.2.6 int Flame::getY ( )

Get Y coordinate

3.8.2.7 std::string Flame::state ( )

Get animation state, returns draw string

3.8.3 Friends And Related Function Documentation

3.8.3.1 bool operator== (Flame & *a,* Point< int > & *b*) [friend]

Equality operator

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

- flame.h
- flame.cpp

3.9 Frame Struct Reference

Public Member Functions

- Frame (std::string iobj, TIME itime)

Public Attributes

- TIME time
- std::string obj

3.9.1 Constructor & Destructor Documentation

3.9.1.1 Frame::Frame (std::string *iobj,* TIME *itime*) [inline]

Constructor

Parameters

iobj	graphic to show
itime	delay after previous

3.9.2 Member Data Documentation

3.9.2.1 std::string Frame::obj

Graphic to show

3.9.2.2 TIME Frame::time

Delay after previous frame

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

- view/animfile.h

3.10 Game Class Reference

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

- game.h
- game.cpp

3.11 GameConfig Class Reference

Public Member Functions

- GameConfig ()

Public Attributes

- int sx
- int sy
- int sb
- bool gen
- bool init
- int nVirus
- int nBomb
- int nFlame

3.11.1 Constructor & Destructor Documentation

3.11.1.1 GameConfig::GameConfig ( ) [inline]

Default constructor

3.11.2 Member Data Documentation

3.11.2.1 bool GameConfig::gen

Generate map

3.11.2.2 bool GameConfig::init

Initialize map

3.11.2.3 int GameConfig::nBomb

Number of bomb size boosters on map

3.11.2.4 int GameConfig::nFlame

Number of flame size boosters on map

3.11.2.5 int GameConfig::nVirus

Number of viruses on map

3.11.2.6 int GameConfig::sb

A squares size on map

3.11.2.7 int GameConfig::sx

Number of columns on map

3.11.2.8 int GameConfig::sy

Number of rows on map

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

- gameconfig.h

3.12 GameManager Class Reference

Public Member Functions

- GameManager (GAME_MODE_TYPE iGameMode, View &iview, Input &iInput, GameTable &iTable)
- void prepareTable ()
- void newRound ()
- void checkState ()
- void checkBefore ()
- void checkAfter ()
- void waitRound (TIME t=1000)
- bool getRun ()
- bool getRound ()

3.12.1 Constructor & Destructor Documentation

3.12.1.1 GameManager::GameManager (GAME_MODE_TYPE *iGameMode,* View & *iview,* Input & *iInput,* GameTable & *iTable*)

Constructor

Parameters

iGameMode	game mode
iview	View class
iInput	Input class
iTable	Table class

3.12.2 Member Function Documentation

3.12.2.1 void GameManager::checkAfter ( )

Check after rounds

3.12.2.2 void GameManager::checkBefore ( )

Check after frames

3.12.2.3 void GameManager::checkState ( )

Check state of game (score of players) and stop if necessary

3.12.2.4 bool GameManager::getRound ( )

End of round

3.12.2.5 bool GameManager::getRun ( )

End of this game

3.12.2.6 void GameManager::newRound ( )

Start new round, reset table

3.12.2.7 void GameManager::prepareTable ( )

Prepare game table

3.12.2.8 void GameManager::waitRound (TIME *t* = 1000)

Wait for round start

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

- gamemanager.h
- gamemanager.cpp

3.13 GameTable Class Reference

Public Member Functions

- GameTable (View &iView)
- ~GameTable ()
- void setPlayers (int n)
- std::string loadMap (std::string file)
- void generateMap ()
- void generateBase ()
- void setStart (int x, int y, int i)
- void spawnPlayer (int i)
- void spawnMonster (int i=-1, int x=-1, int y=-1)
- void spawnWalls (int limit)
- void buildWalls ()
- void move ()
- void setConfig (GameConfig &conf)
- GameConfig & getConfig ()
- const std::vector< Player > & getPlayers () const
- void incScore (int pId)
- int countMonster () const
- void bind (int i, Control *c)
- void execute (NetworkEvent event)
- NetworkEvent getMovement ()

3.13.1 Constructor & Destructor Documentation

3.13.1.1 GameTable::GameTable (View & *iView*)

Constructor

Parameters

iView	View class

3.13.1.2 GameTable::~GameTable ( ) [inline]

Destructor

3.13.2 Member Function Documentation

3.13.2.1 void GameTable::bind (int *i*, Control * *c*) [inline]

Bind player to Control class

Parameters

i	player ID
c	Control class

3.13.2.2 void GameTable::buildWalls ( )

If the wall is build, then put it on the map (after animation)

3.13.2.3 int GameTable::countMonster ( ) const [inline]

Count alive monsters

3.13.2.4 void GameTable::execute (NetworkEvent *event*)

Execute remote command

3.13.2.5 void GameTable::generateBase ( )

Generate an empty map

3.13.2.6 void GameTable::generateMap ( )

Generate a random map

3.13.2.7 GameConfig& GameTable::getConfig ( ) [inline]

Get configuration of map

3.13.2.8 NetworkEvent GameTable::getMovement ( )

Get movement of player for network

3.13.2.9 const std::vector<Player>& GameTable::getPlayers ( ) const [inline]

Get players array (const)

3.13.2.10 void GameTable::incScore (int *pId*) [inline]

Increase score of player

Parameters

pId	player ID

3.13.2.11 std::string GameTable::loadMap (std::string *file*)

Load map from file

Parameters

file	file name

Returns

name of the next map

3.13.2.12 void GameTable::move ( )

Move things on map and check collision

3.13.2.13 void GameTable::setConfig (GameConfig & *conf*)

Set configuration for map

3.13.2.14 void GameTable::setPlayers (int *n*)

Set number of players

3.13.2.15 void GameTable::setStart (int *x*, int *y*, int *i*)

Set start position of player

Parameters

x	coordinate x
y	coordinate y
i	player ID

3.13.2.16 void GameTable::spawnMonster (int *i* = −1, int *x* = −1, int *y* = −1)

Spawn monster. Call it with -1 to get random position or type.

Parameters

i	type of monster
x	coordinate x

y	coordinate y

3.13.2.17 void GameTable::spawnPlayer (int *i*)

Spawn player on map

Parameters

i	player ID

3.13.2.18 void GameTable::spawnWalls (int *limit*)

Start building maximum given number of walls on map

Parameters

limit	maximum number of walls

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

- gametable.h
- gametable.cpp

3.14 Goods Class Reference

Public Member Functions

- Goods ()
- Goods (Point< int > ipoint, GOODS itype)
- int getX ()
- int getY ()
- Point< int > getPoint ()
- GOODS getType ()
- std::string state ()
- std::string effect (Player *p)

3.14.1 Constructor & Destructor Documentation

3.14.1.1 Goods::Goods ( ) [inline]

Default constructor

3.14.1.2 Goods::Goods (Point< int > *ipoint*, GOODS *itype*) [inline]

Constructor

Parameters

ipoint	place of good

itype	type of good (virus, bomb+, flame+)

3.14.2 Member Function Documentation

3.14.2.1 std::string Goods::effect (Player * *p*) [inline]

Apply effect on player

Parameters

p	pointer to player

Returns

text to show

3.14.2.2 Point<int> Goods::getPoint ( ) [inline]

Get point

3.14.2.3 GOODS Goods::getType ( ) [inline]

Get type of good

3.14.2.4 int Goods::getX ( ) [inline]

Get X coordinate

3.14.2.5 int Goods::getY ( ) [inline]

Get Y coordinate

3.14.2.6 std::string Goods::state ( ) [inline]

Get graphic file name of good

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

- goods.h

3.15 Ground Class Reference

Public Member Functions

- Ground ()
- Ground (GROUND_TYPE ig)
- std::string state ()
- Ground & operator= (const GROUND_TYPE &i)

Friends

- bool operator== (const Ground &a, const Ground &b)
- bool operator!= (const Ground &a, const Ground &b)
- std::istream & operator>> (std::istream &input, Ground &ground)
- std::ostream & operator<< (std::ostream &output, const Ground &ground)

3.15.1 Constructor & Destructor Documentation

3.15.1.1 Ground::Ground ( ) [inline]

Default constructor

3.15.1.2 Ground::Ground (GROUND_TYPE *ig*) [inline]

Constructor

Parameters

ig	ground type

3.15.2 Member Function Documentation

3.15.2.1 Ground& Ground::operator= (const GROUND_TYPE & *i*) [inline]

Equality operator

3.15.2.2 std::string Ground::state ( ) [inline]

Graphic file name of current ground

3.15.3 Friends And Related Function Documentation

3.15.3.1 bool operator!= (const Ground & *a*, const Ground & *b*) [friend]

Check not equals

3.15.3.2 std::ostream& operator<< (std::ostream & *output*, const Ground & *ground*) [friend]

Output stream operator

3.15.3.3 bool operator== (const Ground & *a*, const Ground & *b*) [friend]

Check equality

3.15.3.4 std::istream& operator>> (std::istream & *input*, Ground & *ground*) [friend]

Input stream operator

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

- ground.h

3.16 Input Class Reference

Public Member Functions

- Input ()
- ~Input ()
- void updateJoySticks ()
- bool getRun ()
- bool getRunSoft ()
- int getNJoy ()
- int getNPlayers ()
- void setRun (bool irun)
- void setView (View *iView)
- void setMode (bool m)
- Control * operator() (int i)
- void poll (TIME delay)
- void setDelay (TIME delay)

3.16.1 Constructor & Destructor Documentation

3.16.1.1 Input::Input ( )

Constructor, initialize controls

3.16.1.2 Input::~Input ( )

Destructor, release joysticks

3.16.2 Member Function Documentation

3.16.2.1 int Input::getNJoy ( ) [inline]

Get number of joysticks

3.16.2.2 int Input::getNPlayers ( ) [inline]

Get number of players

3.16.2.3 bool Input::getRun ( ) [inline]

Program should run

3.16.2.4 bool Input::getRunSoft ( ) [inline]

Program should step back one level

3.16.2.5 Control* Input::operator() (int *i*) [inline]

Get Control class of selected ID

3.16.2.6 void Input::poll (TIME *delay*)

Poll events with SDL

3.16.2.7 void Input::setDelay (TIME *delay*)

Set delay between frames

3.16.2.8 void Input::setMode (bool *m*) [inline]

Set mode: 0:menu, 1:in-game

3.16.2.9 void Input::setRun (bool *irun*) [inline]

Set run

3.16.2.10 void Input::setView (View * *iView*) [inline]

Set View class

3.16.2.11 void Input::updateJoySticks ( )

Update joysticks, search for newly connected

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

- input.h
- input.cpp

3.17 Logger Class Reference

`#include <logger.h>`

Static Public Member Functions

- static void setLevel (const int _level)
- static void info (const std::string &str)
- static void error (const std::string &str)

3.17.1 Detailed Description

Logger class

3.17.2 Member Function Documentation

3.17.2.1 static void Logger::error (const std::string & *str*) [inline], [static]

Trace error

3.17.2.2 static void Logger::info (const std::string & *str*) [inline], [static]

Trace info

3.17.2.3 static void Logger::setLevel (const int *_level*) [inline], [static]

Set error level

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

- logger.h

3.18 Matrix< T > Class Template Reference

Public Member Functions

- Point< int > find (T &s)
- Matrix ()
- Matrix (int in, int im)
- int getN ()
- int getM ()
- void setSize (int in, int im)
- bool valid (Point< int > p)
- bool valid (int x, int y)
- unsigned int count (const T &e)
- T & operator() (Point< int > p)
- T & operator() (int i, int j)
- const T & operator() (Point< int > p) const
- const T & operator() (int i, int j) const

Friends

- std::istream & operator>> (std::istream &input, Matrix< T > &p)
- std::ostream & operator<< (std::ostream &output, const Matrix< T > &p)

3.18.1 Constructor & Destructor Documentation

3.18.1.1 template<typename T> Matrix< T >::Matrix ( ) [inline]

Default constructor

3.18.1.2 template<typename T> Matrix< T >::Matrix (int *in,* int *im*) [inline]

Constructor

Parameters

in	number of rows
im	number of columns

3.18.2 Member Function Documentation

3.18.2.1 template<typename T> unsigned int Matrix< T >::count (const T & *e*) [inline]

Count an element's number of occurences in matrix

3.18.2.2 template<typename T> Point<int> Matrix< T >::find (T & *s*) [inline]

Find the first occurance of element in matrix

3.18.2.3 template<typename T> int Matrix< T >::getM ( ) [inline]

Number of columns

3.18.2.4 template<typename T> int Matrix< T >::getN ( ) [inline]

Number of rows

3.18.2.5 template<typename T> T& Matrix< T >::operator() (Point< int > *p*) [inline]

Get element at point

3.18.2.6 template<typename T> T& Matrix< T >::operator() (int *i*, int *j*) [inline]

Get element at coordinates

3.18.2.7 template<typename T> const T& Matrix< T >::operator() (Point< int > *p*) const [inline]

Get element at point (const)

3.18.2.8 template<typename T> const T& Matrix< T >::operator() (int *i*, int *j*) const [inline]

Get element at coordinates (const)

3.18.2.9 template<typename T> void Matrix< T >::setSize (int *in*, int *im*) [inline]

Set size of matrix

3.18.2.10 template<typename T> bool Matrix< T >::valid (Point< int > *p*) [inline]

The given point is in a valid range or not

3.18.2.11 template<typename T> bool Matrix< T >::valid (int *x*, int *y*) [inline]

The given coordinates is in a valid range or not

3.18.3 Friends And Related Function Documentation

3.18.3.1 template<typename T> std::ostream& operator<< (std::ostream & *output,* const Matrix< T > & *p*) [friend]

Output stream operator

3.18.3.2 template<typename T> std::istream& operator>> (std::istream & *input,* Matrix< T > & *p*) [friend]

Input stream operator

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

- view/matrix.h

3.19 Menu Class Reference

Public Member Functions

- Menu (View &iView, Input &iInput)
- GAME_MODE_TYPE getMode ()
- bool mainLoop ()
- void down ()
- void up ()
- void check ()

3.19.1 Constructor & Destructor Documentation

3.19.1.1 Menu::Menu (View & *iView,* Input & *iInput*)

Constructor

Parameters

iView	View class
iInput	Input class

3.19.2 Member Function Documentation

3.19.2.1 void Menu::check ( )

Check time between the last change of menu items and now, step to next if necessary

3.19.2.2 void Menu::down ( )

Go down one item

3.19.2.3 GAME_MODE_TYPE Menu::getMode ( )

Get selected game mode

3.19.2.4 bool Menu::mainLoop ( )

Main loop, draws on screen and checks input

3.19.2.5 void Menu::up ( )

Go up one item

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

- menu.h
- menu.cpp

3.20 MenuItem Class Reference

Public Member Functions

- MenuItem (std::string inp, int inext, GAME_MODE_TYPE imode)

Public Attributes

- int next
- GAME_MODE_TYPE mode
- std::string txt

3.20.1 Constructor & Destructor Documentation

3.20.1.1 MenuItem::MenuItem (std::string *inp*, int *inext*, GAME_MODE_TYPE *imode*) [inline]

Constructor

3.20.2 Member Data Documentation

3.20.2.1 GAME_MODE_TYPE MenuItem::mode

Game mode

3.20.2.2 int MenuItem::next

Next menu item when selected

3.20.2.3 std::string MenuItem::txt

Text to show

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

- menu.h

3.21 Monster Class Reference

`#include <monster.h>`

Public Member Functions

- Monster (int itype, int isb, Point< float > ipoint)
- bool isActive ()
- bool collide ()
- std::string die ()
- bool alive ()
- void move (TIME now)
- bool change ()
- void setX (float x)
- void setY (float y)
- void setDX (float dx)
- void setDY (float dy)
- Point< float > getPoint ()
- Point< int > getPointInt ()
- float getX ()
- float getY ()
- std::string state ()
- bool getWall ()

3.21.1 Detailed Description

This class contains a monster

3.21.2 Constructor & Destructor Documentation

3.21.2.1 Monster::Monster (int *itype*, int *isb*, Point< float > *ipoint*)

Constructor

Parameters

itype	type of monster
isb	size of a brick on map, necessary for random way changes
ipoint	starting point

3.21.3 Member Function Documentation

3.21.3.1 bool Monster::alive ( )

Is alive

3.21.3.2 bool Monster::change ( )

Change direction

3.21.3.3 bool Monster::collide ( )

Is colliding with wall

3.21.3.4 std::string Monster::die ( )

Dies

Returns

animation of dying

3.21.3.5 Point< float > Monster::getPoint ( )

Get exact point on screen

3.21.3.6 Point< int > Monster::getPointInt ( )

Get rounded point, which rectangle is the nearest

3.21.3.7 bool Monster::getWall ( )

Return that the monster can walk through walls or not

3.21.3.8 float Monster::getX ( )

Get exact X coordinate

3.21.3.9 float Monster::getY ( )

Get exact Y coordinate

3.21.3.10 bool Monster::isActive ( )

Not yet active, false for a few seconds when the class is created

3.21.3.11 void Monster::move (TIME *now*)

Move the monster, with calculating the difference between current and previous frame time

Parameters

now	current frame time

3.21.3.12 void Monster::setDX (float *dx*)

Set direction X

3.21.3.13 void Monster::setDY (float *dy*)

Set direction Y

3.21.3.14 void Monster::setX (float *x*)

Set X coordinate

3.21.3.15 void Monster::setY (float *y*)

Set Y coordinate

3.21.3.16 std::string Monster::state ( )

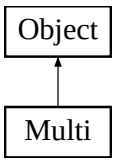
Get animation state

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

- monster.h
- monster.cpp

3.22 Multi Class Reference

Inheritance diagram for Multi:



Public Member Functions

- Multi ()
- Multi (std::string fileName, std::map< std::string, Object * > *igraphs)
- void draw (float x, float y, float w, float h, std::string param)

3.22.1 Constructor & Destructor Documentation

3.22.1.1 Multi::Multi ( ) [inline]

Default constructor

3.22.1.2 Multi::Multi (std::string *fileName,* std::map< std::string, Object * > * *igraphs*) [inline]

Constructor

Parameters

fileName	name of multi file (.mlt)
igraphs	list of loaded graphics from View class

3.22.2 Member Function Documentation

3.22.2.1 void Multi::draw (float *x,* float *y,* float *w,* float *h,* std::string *param*) [virtual]

Draw a specific slice of a Sprite

Parameters

x	coordinate
y	coordinate
w	width
h	height
param	full string param, determines which slice need to be drawn

Reimplemented from Object.

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

- view/multi.h
- view/multi.cpp

3.23 Network Class Reference

Public Member Functions

- Network ()
- ~Network ()
- void getInternetList ()
- void startListen ()
- void stopListen ()
- void mainLoop ()

3.23.1 Constructor & Destructor Documentation

3.23.1.1 Network::Network ( )

Constructor

3.23.1.2 Network::~Network ( )

Destructor

3.23.2 Member Function Documentation

3.23.2.1 void Network::getInternetList ( )

Get list of Internet servers

3.23.2.2 void Network::mainLoop ( )

Main loop, for doing the work

3.23.2.3 void Network::startListen ( )

Start server

3.23.2.4 void Network::stopListen ( )

Stop server

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

- network.h
- network.cpp

3.24 NetworkEvent Struct Reference

Public Attributes

- EVENT_TYPE e
- int id
- float x
- float y

3.24.1 Member Data Documentation

3.24.1.1 EVENT_TYPE NetworkEvent::e

Type of event

3.24.1.2 int NetworkEvent::id

ID of player

3.24.1.3 float NetworkEvent::x

Coordinate X

3.24.1.4 float NetworkEvent::y

Coordinate Y

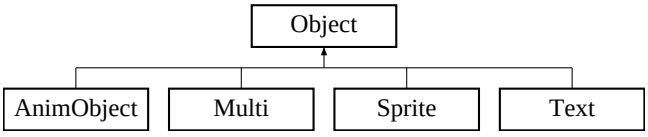
The documentation for this struct was generated from the following file:

- networkevent.h

3.25 Object Class Reference

`#include <object.h>`

Inheritance diagram for Object:



Public Member Functions

- virtual void draw (float x, float y, float w, float h, std::string param="")
- virtual void bind ()
- virtual ~Object ()

3.25.1 Detailed Description

Base class of drawable objects

3.25.2 Constructor & Destructor Documentation

3.25.2.1 virtual Object::~Object ( ) [inline], [virtual]

Destructor

3.25.3 Member Function Documentation

3.25.3.1 virtual void Object::bind ( ) [inline], [virtual]

Bind graphic to draw texture

Reimplemented in Sprite.

3.25.3.2 virtual void Object::draw (float *x*, float *y*, float *w*, float *h*, std::string *param* = " ") [inline], [virtual]

Drawing object on screen

Parameters

x	coordinate
y	coordinate
w	width
h	height
param	full drawing string

Reimplemented in Sprite, Text, AnimObject, and Multi.

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

- view/object.h

3.26 Player Class Reference

`#include <player.h>`

Public Member Functions

- Player ()
- ~Player ()
- void init ()
- void novirus ()
- void move (TIME inow)
- void move (Point< float > p)

- int infect ()
- int infect (int type)
- void die ()
- void win ()
- std::string state ()
- bool getPut ()
- int getId () const
- Control * getControl ()
- Point< float > getPoint ()
- float getX ()
- float getY ()
- float getPrevX ()
- float getPrevY ()
- int getBombX ()
- int getBombY ()
- int getSize ()
- int getBombs ()
- int getLife ()
- bool alive () const
- int getScore () const
- int getKill () const
- int getKilled () const
- std::string getSkin () const
- TIME getDie () const
- TIME getShowKills () const
- void setId (int i)
- void setControl (Control *icontrol)
- void setX (float x)
- void setY (float y)
- void setBombX (int x)
- void setBombY (int y)
- void incSize ()
- void incBombs ()
- void decBombs ()
- void setLife (int l)
- void setPut (bool p)
- void setScore (int s)
- void incScore ()
- void setKill (int k)
- void incKill ()
- void incKill (TIME iShowKills)
- void decKill ()
- void decKill (TIME iShowKills)
- void setShowKills (TIME iShowKills)
- void setKilled (int k)
- void incKilled ()
- void setSkin (std::string iSkin)
- void setDie (TIME idie)
- void setStart (Point< float > p)
- void setStart ()

3.26.1 Detailed Description

This class contains all information about a player

3.26.2 Constructor & Destructor Documentation

3.26.2.1 Player::Player ( )

Constructor, initialize player

3.26.2.2 Player::~Player ( )

Destructor

3.26.3 Member Function Documentation

3.26.3.1 bool Player::alive ( ) const [inline]

Return if the player is alive or not

3.26.3.2 void Player::decBombs ( ) [inline]

Decrease number of bombs

3.26.3.3 void Player::decKill ( ) [inline]

Decrease kills (when suicide)

3.26.3.4 void Player::decKill (TIME *iShowKills*) [inline]

Decrease kills (when suicide) with showing the statistics

Parameters

iShowKills	frame time when it should disappear

3.26.3.5 void Player::die ( )

Player dies

3.26.3.6 int Player::getBombs ( ) [inline]

Get number of bombs

3.26.3.7 int Player::getBombX ( ) [inline]

Get the X coordinate where the player put bomb

3.26.3.8 int Player::getBombY ( ) [inline]

Get the Y coordinate where the player put bomb

3.26.3.9 Control* Player::getControl ( ) [inline]

Returns the Control class of player

3.26.3.10 TIME Player::getDie ( ) const [inline]

Return the frame time when player died

3.26.3.11 int Player::getId ( ) const [inline]

Return the ID of the player

3.26.3.12 int Player::getKill ( ) const [inline]

Get the number of kills

3.26.3.13 int Player::getKilled ( ) const [inline]

Get the number of get killed

3.26.3.14 int Player::getLife ( ) [inline]

Get life of player

3.26.3.15 Point<float> Player::getPoint ( ) [inline]

Get position of player

3.26.3.16 float Player::getPrevX ( ) [inline]

Get previous X coordinate of player

3.26.3.17 float Player::getPrevY ( ) [inline]

Get previous Y coordinate of player

3.26.3.18 bool Player::getPut ( )

Return that the player puts bomb or not

3.26.3.19 int Player::getScore ( ) const [inline]

Get score of player

3.26.3.20 TIME Player::getShowKills ( ) const [inline]

Return when showing kill statistic should disappear

3.26.3.21 int Player::getSize ( ) [inline]

Get flame size

3.26.3.22 std::string Player::getSkin ( ) const [inline]

Get skin of player (prefix for graphic files)

3.26.3.23 float Player::getX ( ) [inline]

Get X coordinate of player

3.26.3.24 float Player::getY ( ) [inline]

Get Y coordinate of player

3.26.3.25 void Player::incBombs ( ) [inline]

Increase number of bombs

3.26.3.26 void Player::incKill ( ) [inline]

Increase kills

3.26.3.27 void Player::incKill (TIME *iShowKills*) [inline]

Increase kills with showing the statistics

Parameters

iShowKills	frame time when it should disappear

3.26.3.28 void Player::incKilled ( ) [inline]

Increase killed by other

3.26.3.29 void Player::incScore ( ) [inline]

Increase score

3.26.3.30 void Player::incSize ( ) [inline]

Increase flame size

3.26.3.31 int Player::infect ( )

Infect player with random virus

3.26.3.32 int Player::infect (int *type*)

Infect player with specific virus

3.26.3.33 void Player::init ( )

Set values to default, including position and anything that effected the player in previous rounds

3.26.3.34 void Player::move (TIME *inow*)

Move the player to the direction given by Control class, with calculating the time difference, so the frame rate doesn't matter. For local players.

Parameters

inow	current frame time

3.26.3.35 void Player::move (Point< float > *p*)

Move the player to a specific point. For remote players.

Parameters

p	point where it moves

3.26.3.36 void Player::novirus ( )

Reset virus attributes

3.26.3.37 void Player::setBombX (int *x*) [inline]

Set X coordinate for the last bomb put

3.26.3.38 void Player::setBombY (int *y*) [inline]

Set Y coordinate for the last bomb put

3.26.3.39 void Player::setControl (Control * *icontrol*) [inline]

Set Control class for player, only one can be active per player

3.26.3.40 void Player::setDie (TIME *idie*) [inline]

Set frame time when the player died

3.26.3.41 void Player::setId (int *i*) [inline]

Set id of player

3.26.3.42 void Player::setKill (int *k*) [inline]

Set kills

3.26.3.43 void Player::setKilled (int *k*) [inline]

Set killed by other

3.26.3.44 void Player::setLife (int *l*) [inline]

Set life

3.26.3.45 void Player::setPut (bool *p*) [inline]

Set put

3.26.3.46 void Player::setScore (int *s*) [inline]

Set score

3.26.3.47 void Player::setShowKills (TIME *iShowKills*) [inline]

Set end time of showing statistics

3.26.3.48 void Player::setSkin (std::string *iSkin*) [inline]

Set skin of player

3.26.3.49 void Player::setStart (Point< float > *p*) [inline]

Set the start point of player

3.26.3.50 void Player::setStart ( ) [inline]

Position the player to the start point already set

3.26.3.51 void Player::setX (float *x*) [inline]

Set X coordinate

3.26.3.52 void Player::setY (float *y*) [inline]

Set Y coordinate

3.26.3.53 std::string Player::state ( )

Returns the current animation phase of player

Returns

graphic name

3.26.3.54 void Player::win ( )

Player wins

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

- player.h
- player.cpp

3.27 Point< T > Class Template Reference

Public Member Functions

- Point ()
- Point (T ix, T iy)
- T getX () const
- T getY () const
- void setX (T ix)
- void setY (T iy)
- Point< T > & operator= (const Point< T > &a)

Friends

- bool operator== (const Point< T > &a, const Point< T > &b)
- T distance (const Point< T > &a, const Point< T > &b)
- WAY Way (const Point< T > &a, const Point< T > &b)

3.27.1 Constructor & Destructor Documentation

3.27.1.1 template<typename T> Point< T >::Point ( ) [inline]

Default constructor

3.27.1.2 template<typename T> Point< T >::Point (T *ix*, T *iy*) [inline]

Constructor

Parameters

ix	coordinate x
iy	coordinate y

3.27.2 Member Function Documentation

3.27.2.1 template<typename T> T Point< T >::getX ( ) const [inline]

Getter for X coordinate

3.27.2.2 template<typename T> T Point< T >::getY ( ) const [inline]

Getter for Y coordinate

3.27.2.3 template<typename T> Point<T>& Point< T >::operator= (const Point< T > & *a*) [inline]

Equals operator

Parameters

a	input

Returns

the class

3.27.2.4 template<typename T> void Point< T >::setX (T *ix*) [inline]

Setter for X coordinate

3.27.2.5 template<typename T> void Point< T >::setY (T *iy*) [inline]

Setter for Y coordinate

3.27.3 Friends And Related Function Documentation

3.27.3.1 template<typename T> T distance (const Point< T > & *a*, const Point< T > & *b*) [friend]

Distance between two points

3.27.3.2 template<typename T> bool operator== (const Point< T > & *a*, const Point< T > & *b*) [friend]

Equality check

3.27.3.3 template<typename T> WAY Way (const Point< T > & *a*, const Point< T > & *b*) [friend]

A special function, calculates the direction of a vector

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

- point.h

3.28 Rect Struct Reference

`#include <rect.h>`

Public Member Functions

- Rect ()
- Rect (float ix, float iy, float iw, float ih)

Public Attributes

- float x
- float y
- float w
- float h

3.28.1 Detailed Description

This class stores a rectangle with float coordinates and dimensions

3.28.2 Constructor & Destructor Documentation

3.28.2.1 Rect::Rect ( ) [inline]

Default constructor

3.28.2.2 Rect::Rect (float *ix,* float *iy,* float *iw,* float *ih*) [inline]

Constructor

Parameters

ix	coordinate
iy	coordinate
iw	width
ih	height

3.28.3 Member Data Documentation

3.28.3.1 float Rect::h

Height

3.28.3.2 float Rect::w

Width

3.28.3.3 float Rect::x

Coordinate X

3.28.3.4 float Rect::y

Coordinate Y

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

- view/rect.h

3.29 ScoreBoard Class Reference

Public Member Functions

- ScoreBoard (View &iView, Input &iInput, GameTable &iTable)
- bool mainLoop ()

3.29.1 Constructor & Destructor Documentation

3.29.1.1 ScoreBoard::ScoreBoard (View & *iView,* Input & *iInput,* GameTable & *iTable*)

Constructor

Parameters

iView	View class
iInput	Input class
iTable	Table class

3.29.2 Member Function Documentation

3.29.2.1 bool ScoreBoard::mainLoop ( )

Draws the scoreboard on screen after a round

Returns

is not finished

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

- scoreboard.h
- scoreboard.cpp

3.30 Select Class Reference

`#include <select.h>`

Public Member Functions

- Select (View &iview, Input &iinput, GameTable &itable)
- bool mainLoop ()
- bool getRun ()
- int getN ()

3.30.1 Detailed Description

This class allows the users to select controllers before starting the game

3.30.2 Constructor & Destructor Documentation

3.30.2.1 Select::Select (View & *iview,* Input & *iinput,* GameTable & *itable*)

Constructor

Parameters

iview	View class
iinput	Input class
itable	Table class

3.30.3 Member Function Documentation

3.30.3.1 int Select::getN ( ) [inline]

Getter for number of players

3.30.3.2 bool Select::getRun ( ) [inline]

Getter for run

3.30.3.3 bool Select::mainLoop ( )

The main loop, shows the menu for selecting input

Returns

selecting is not finished

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

- select.h
- select.cpp

3.31 Server Class Reference

Public Member Functions

- Server ()
- ~Server ()
- void mainLoop ()

3.31.1 Constructor & Destructor Documentation

3.31.1.1 Server::Server ( )

Constructor

3.31.1.2 Server::~Server ( )

Destructor

3.31.2 Member Function Documentation

3.31.2.1 void Server::mainLoop ( )

Main loop of server, does everything

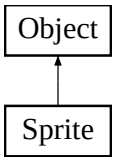
The documentation for this class was generated from the following files:

- server.h
- server.cpp

3.32 Sprite Class Reference

`#include <sprite.h>`

Inheritance diagram for Sprite:



Public Member Functions

- Sprite ()
- Sprite (std::string fileName)
- ~Sprite ()
- void draw (float x, float y, float w, float h, std::string param="")
- void bind ()

3.32.1 Detailed Description

Class for handle single drawable objects.

3.32.2 Constructor & Destructor Documentation

3.32.2.1 Sprite::Sprite ( ) [inline]

Default constructor

3.32.2.2 Sprite::Sprite (std::string *fileName*) [inline]

Constructor

Parameters

fileName	name of graphic file (.png)

3.32.2.3 Sprite::~Sprite ( )

Destructor, free memory

3.32.3 Member Function Documentation

3.32.3.1 void Sprite::bind ( ) [inline], [virtual]

Bind graphic for drawing

Reimplemented from Object.

3.32.3.2 void Sprite::draw (float *x*, float *y*, float *w*, float *h*, std::string *param* = " ") [virtual]

Draw sprite on screen

Parameters

x	coordinate
y	coordinate
w	width
h	height
param	full draw string

Reimplemented from Object.

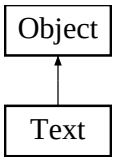
The documentation for this class was generated from the following files:

- view/sprite.h
- view/sprite.cpp

3.33 Text Class Reference

`#include <text.h>`

Inheritance diagram for Text:



Public Member Functions

- Text ()
- Text (std::string fileName, std::map< std::string, Object * > *igraphs)
- void draw (float x, float y, float w, float h, std::string param)

3.33.1 Detailed Description

Class for text handling using Multi class

3.33.2 Constructor & Destructor Documentation

3.33.2.1 Text::Text ( ) [inline]

Default constructor

3.33.2.2 Text::Text (std::string *fileName*, std::map< std::string, Object * > * *igraphs*) [inline]

Constructor, load resources

Parameters

fileName	name of txh file

igraphs	graphics from view class

3.33.3 Member Function Documentation

3.33.3.1 void Text::draw (float *x,* float *y,* float *w,* float *h,* std::string *param*) [virtual]

Draw text (font.txh::my_text)

Reimplemented from Object.

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

- view/text.h
- view/text.cpp

3.34 Translate Class Reference

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

- translate.h

3.35 Utf8 Class Reference

`#include <utf8.h>`

Public Member Functions

- Utf8 ()
- bool notSpace ()
- bool notNull ()
- bool isBr ()

Friends

- std::ostream & operator<< (std::ostream &output, const Utf8 &u)
- std::istream & operator>> (std::istream &input, Utf8 &u)
- bool operator== (const Utf8 &u, const Utf8 &v)
- bool operator> (const Utf8 &u, const Utf8 &v)
- bool operator< (const Utf8 &u, const Utf8 &v)

3.35.1 Detailed Description

Class for UTF-8 character handling

3.35.2 Constructor & Destructor Documentation

3.35.2.1 Utf8::Utf8 ( ) [inline]

Constructor

3.35.3 Member Function Documentation

3.35.3.1 bool Utf8::isBr ( ) [inline]

Character is not line-break

3.35.3.2 bool Utf8::notNull ( ) [inline]

Character is not null

3.35.3.3 bool Utf8::notSpace ( ) [inline]

Character is not space

3.35.4 Friends And Related Function Documentation

3.35.4.1 bool operator< (const Utf8 & *u,* const Utf8 & *v*) [friend]

Less than operator

3.35.4.2 std::ostream& operator<< (std::ostream & *output,* const Utf8 & *u*) [friend]

Output operator, don't write second byte if it is null

Returns

standard ostream

3.35.4.3 bool operator== (const Utf8 & *u,* const Utf8 & *v*) [friend]

Equality operator

3.35.4.4 bool operator> (const Utf8 & *u,* const Utf8 & *v*) [friend]

Greater than operator

3.35.4.5 std::istream& operator>> (std::istream & *input,* Utf8 & *u*) [friend]

Input operator, smartly determines UTF-8 characters and only read the second byte if it is valid UTF-8 character

Returns

standard istream

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

- view/utf8.h

3.36 View Class Reference

`#include <view.h>`

Public Member Functions

- View ()
- ~View ()
- void swap ()
- bool load (std::string fileName)
- bool loadPackage (std::string fileName)
- void draw (Rect r, std::string res)
- void draw (float x, float y, float w, float h, std::string res)
- void resize_event (int &w, int &h)
- void toggleFullScreen ()
- void drawAnim ()
- TIME getNow ()
- void setNow ()
- int getW ()
- int getH ()
- void setTickInterval (int i)
- void Screenshot ()
- void Toggle ()
- void frameBegin ()
- void frameEnd ()
- TIME frameDelay ()

3.36.1 Detailed Description

Main class for handling graphics

3.36.2 Constructor & Destructor Documentation

3.36.2.1 View::View ( )

Constructor, initialize screen

3.36.2.2 View::~View ( )

Destructor, unload objects

3.36.3 Member Function Documentation

3.36.3.1 void View::draw (Rect *r*, std::string *res*)

Draw on frame

Parameters

r	rectangle of drawing position
res	full drawing string

3.36.3.2 void View::draw (float *x*, float *y*, float *w*, float *h*, std::string *res*)

Draw on frame

Parameters

x	coordinate
y	coordinate
w	width
h	height
res	full drawing string

3.36.3.3 void View::drawAnim ( )

Draw animations on frame, delete ended animations

3.36.3.4 void View::frameBegin ( )

Begin of frame, actualilze time variables for current and previous frame

3.36.3.5 TIME View::frameDelay ( )

Amount of sleep needed for enough frames per second

3.36.3.6 void View::frameEnd ( )

End of frame, sleep for a while to save CPU time

3.36.3.7 int View::getH ( ) [inline]

Get screen height

3.36.3.8 TIME View::getNow ( ) [inline]

Get current frame time

3.36.3.9 int View::getW ( ) [inline]

Get screen width

3.36.3.10 bool View::load (std::string *fileName*)

Load image, animation or text file

Parameters

fileName	name of file, extension determines the type

Returns

success

3.36.3.11 bool View::loadPackage (std::string *fileName*)

Load resources from list

Parameters

fileName	name of package

Returns

success

3.36.3.12 void View::resize_event (int & *w,* int & *h*)

Resize callback function

Parameters

w	width
h	height

3.36.3.13 void View::Screenshot ( )

Request to take screenshot at the end of frame

3.36.3.14 void View::setNow ( ) [inline]

Set current frame time to actual time

3.36.3.15 void View::setTickInterval (int *i*) [inline]

Set refresh rate

3.36.3.16 void View::swap ( )

Swap buffers, draw on screen

3.36.3.17 void View::Toggle ( ) [inline]

Request to toggle fullscreen

3.36.3.18 void View::toggleFullScreen ( )

Toggle fullscreen

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

- view/view.h
- view/view.cpp

Index