OWNSUBJECT5

1.0

Generated by Doxygen 1.6.3

Tue Dec 14 21:12:12 2010

Contents

Chapter 1

Data Structure Index

1.1 Class Hierarchy

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

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

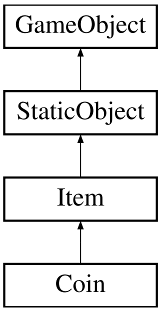
Data Structure Documentation

3.1 Coin Class Reference

A header for coins.

`#include <coin.hh>`

Inheritance diagram for Coin:



Public Member Functions

- Coin (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef)

  A constructor.

- Coin (const Coin &other)

  Copy-constructor.

- Coin & operator= (const Coin &other)

  assignment operator.

- ∼Coin ()

  A destructor.

- void gather ()

  gathers the coin.

3.1.1 Detailed Description

A header for coins.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Coin::Coin (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*)

A constructor.

A normal class for coins.

Parameters

init_x initial x.

init_y initial y.

gameIO reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

coin class is example of item subclasses A constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO reference to gameio struct.

gameGraphicRef reference to gamegraphic struct.

3.1.2.2 Coin::Coin (const Coin & *other*)

Copy-constructor.

Parameters

other reference to the object to be copied.

3.1.3 Member Function Documentation

3.1.3.1 Coin & Coin::operator= (const Coin & *other*)

assignment operator.

Parameters

other reference to player-object to be assigned.

Returns

this.

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

- coin.hh
- coin.cc

3.2 Core Class Reference

Core class.

Public Member Functions

- Core (GameIO &gameio, GameGraphic &gg)

  Constructor.

- void start ()

  Start the core program.

- void start_new_game ()
- void continue_game ()

3.2.1 Detailed Description

Core class. Core class does the core functionality of the game.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Core::Core (GameIO & *gameio*, GameGraphic & *gg*) `[inline]`

Constructor.

Sets game IO and graphics, sounds and music for the game.

Parameters

gameio Game IO.
gg Game graphics and sounds.

3.2.3 Member Function Documentation

3.2.3.1 void Core::continue_game () `[inline]`

Continue the on-going game.

3.2.3.2 void Core::start () `[inline]`

Start the core program.

Starts the core program. Draws the menu and waits for a menu selection.

3.2.3.3 void Core::start_new_game () `[inline]`

Starts a new game.

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

- mario.cc

3.3 Game Class Reference

Game logic.

```
#include <game.hh>
```

Public Member Functions

- Game (Options &options, GameIO &gameio, GameGraphic &gameGraphic)

  Constructs a new Game.

- Game (const Game &other)

  Copy-constructor.

- ∼Game ()

  Destructor.

- void init_tiles ()
- void init_monsters ()

  Generates a new monster to the field when the time is right.

- void pause ()

  Sets the game paused.

- bool isPaused ()

  Function to find out if game is paused or not.

- bool isStarted ()

  Function to find out if game is paused or not.

- bool isOver ()

  Function to find out if game is paused or not.

- void handleInput (float t)

  Handles the input from the users.

- void moveMonsters (float time, float totalTime)

  Moves all the monsters.

- void addGravity (float t, MovingObject &mo)

  Adds acceleration down if object is in the air.

- void movePlayers (float t)

  Moves players according to user input.

- void checkCollisionToTiles (MovingObject &mo)

  Check the collisions to tiles.

- void drawTiles ()

  Draws all the Tiles stored to vector 'tiles'.

- void drawPipes ()

  Draws pipes to the game field.

- void drawMonsters ()

  Draws monsters to the game field.

- void drawShrooms ()

  Draws shrooms currently on the game field.

- void runGame ()

  Runs the game.

- Game & operator= (const Game &other)

  assignment operator.

- void checkCollisions (Player &p1)

  Checks the collisions to of given player to other elements of the game.

- void drawCoins ()

  Draws coins.

- void drawGatheredCoins ()

  Draws the coins player have gathered to the up bar of the window.

3.3.1 Detailed Description

Game logic.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Game::Game (Options & *options*, GameIO & *gameio*, GameGraphic & *gameGraphic*)

Constructs a new Game.

Parameters

options Game options for this game.

gameio GameIO object containing the I/O objects needed in the game.

gameGraphic Struct for the graphical part used in the game.

3.3.2.2 Game::Game (const Game & *other*)

Copy-constructor.

Parameters

other Game to copy.

3.3.2.3 Game::∼Game ()

Destructor.

Frees all the resources initialized in the constructor.

3.3.3 Member Function Documentation

3.3.3.1 void Game::addGravity (float *t*, MovingObject & *mo*)

Adds acceleration down if object is in the air.

Parameters

t Time

mo Object.

t Time.

mo Object.

3.3.3.2 void Game::checkCollisionToTiles (MovingObject & *mo*)

Check the collisions to tiles.

Parameters

mo Object to check.

3.3.3.3 void Game::handleInput (float *t*)

Handles the input from the users.

Parameters

t Time.

3.3.3.4 void Game::init_monsters ()

Generates a new monster to the field when the time is right.

Initializes monsters of the game.

3.3.3.5 void Game::init_tiles ()

Initializes tiles.

3.3.3.6 bool Game::isOver ()

Function to find out if game is paused or not.

Returns

false if game is initialized but never started, true otherwise.

3.3.3.7 bool Game::isPaused ()

Function to find out if game is paused or not.

Returns

true if game is pause, false otherwise.

3.3.3.8 bool Game::isStarted ()

Function to find out if game is paused or not.

Returns

false if game is initialized but never started, true otherwise.

3.3.3.9 void Game::moveMonsters (float *time*, float *total_time*)

Moves all the monsters.

Moving is done by calling move() function of each monster one by one.

Parameters

time Time.
total_time Total time passed from the beginning of the game.

3.3.3.10 void Game::movePlayers (float *t*)

Moves players according to user input.

Parameters

t Time.

3.3.3.11 void Game::pause ()

Sets the game paused.

Sets the game paused, used e.g. if we enter the menu.

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

- game.hh
- game.cc

3.4 GameGraphic Struct Reference

GameGraphic struct.

`#include <gamegraphic.hh>`

Public Member Functions

- GameGraphic (GameIO &gameio)

  GameGraphic constructor.
- GameGraphic (GameGraphic &other)

  Copy-constructor.
- GameGraphic & operator= (GameGraphic &other)

  Assignment-operator.
- ∼GameGraphic ()

  Destructor.
- CL_Sprite getStillFabioSprite ()

  Getter for fabio sprite.
- CL_Sprite getStillSuperFabioSprite ()

  Getter for fabio sprite.
- CL_Sprite getMovingFabioSprite ()

  Getter for fabio sprite.
- CL_Sprite getMovingSuperFabioSprite ()

  Getter for fabio sprite.
- CL_Sprite getJumpingFabioSprite ()

  Getter for fabio sprite.
- CL_Sprite getJumpingSuperFabioSprite ()

  Getter for fabio sprite.
- CL_Sprite getStillUoleviSprite ()

  Getter for uolevi sprite.
- CL_Sprite getStillSuperUoleviSprite ()

  Getter for uolevi sprite.
- CL_Sprite getMovingUoleviSprite ()

  Getter for uolevi sprite.
- CL_Sprite getMovingSuperUoleviSprite ()

  Getter for uolevi sprite.

- CL_Sprite getJumpingUoleviSprite ()

    Getter for uolevi sprite.

- CL_Sprite getJumpingSuperUoleviSprite ()

    Getter for uolevi sprite.

- CL_Sprite getStillGoombaSprite ()

    Getter for goomba sprite.

- CL_Sprite getMovingGoombaSpriteL ()

    Getter for goomba sprite.

- CL_Sprite getMovingGoombaSpriteR ()

    Getter for goomba sprite.

- CL_Sprite getStillKoopaSprite ()

    Getter for koopa sprite.

- CL_Sprite getMovingKoopaSpriteL ()

    Getter for koopa sprite.

- CL_Sprite getMovingKoopaSpriteR ()

    Getter for koopa.

- CL_Sprite getStillTileSprite ()

    Getter for koopa sprite.

- CL_Sprite getMovingTileSprite ()

    Getter for tile sprite.

- CL_Sprite getShroomSprite ()

    Getter for shroom sprite.

- CL_Sprite getCoinSprite ()

    Getter for coin sprite.

- CL_Texture & getBackground ()

    Getter for background texture.

- CL_Texture & getMenuBackground ()

    Getter for menu background texture.

- CL_Sprite getLeftPipeSprite ()

    Getter for pipe sprite.

- CL_Sprite getRightPipeSprite ()

    Getter for pipe sprite.

- CL_Sprite getDeadSprite ()

    Getter for dead sprite.

- CL_SoundBuffer_Session getBgMusic ()

  Getter for background music.

- CL_SoundBuffer_Session getMenuBgMusic ()

  Getter for menu background music.

- std::vector< CL_SoundBuffer_Session > getFx ()

  Getter for effect sounds.

- std::vector< CL_SoundBuffer_Session > getMenuFx ()

  Getter for menu effect sounds.

3.4.1 Detailed Description

GameGraphic struct. GameGraphic struct contains and initializes all necessary graphics and sounds for the game.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 GameGraphic::GameGraphic (GameIO & *gameio*) `[inline]`

GameGraphic constructor.

Initializes all necessary graphics and sounds. All paths for graphics and sound are hard-coded here.

Parameters

gameio GameIO object containing the I/O objects needed in the game.

3.4.2.2 GameGraphic::GameGraphic (GameGraphic & *other*) `[inline]`

Copy-constructor.

Parameters

other reference to the object to be copied.

3.4.2.3 GameGraphic::∼GameGraphic () `[inline]`

Destructor.

RO3.

3.4.3 Member Function Documentation

3.4.3.1 CL_Texture& GameGraphic::getBackground () `[inline]`

Getter for background texture.

Returns

background texture.

3.4.3.2 CL_SoundBuffer_Session GameGraphic::getBgMusic () `[inline]`

Getter for background music.

Returns

background music.

3.4.3.3 CL_Sprite GameGraphic::getCoinSprite () `[inline]`

Getter for coin sprite.

Returns

coin.

3.4.3.4 CL_Sprite GameGraphic::getDeadSprite () `[inline]`

Getter for dead sprite.

Returns

dead sprite.

3.4.3.5 std::vector<CL_SoundBuffer_Session> GameGraphic::getFx () `[inline]`

Getter for effect sounds.

Returns

effect sounds.

3.4.3.6 CL_Sprite GameGraphic::getJumpingFabioSprite () `[inline]`

Getter for fabio sprite.

Returns

fabio sprite.

3.4.3.7 CL_Sprite GameGraphic::getJumpingSuperFabioSprite () `[inline]`

Getter for fabio sprite.

Returns

fabio sprite.

3.4.3.8 CL_Sprite GameGraphic::getJumpingSuperUoleviSprite () `[inline]`

Getter for uolevi sprite.

Returns

uolevi sprite.

3.4.3.9 CL_Sprite GameGraphic::getJumpingUoleviSprite () `[inline]`

Getter for uolevi sprite.

Returns

uolevi sprite.

3.4.3.10 CL_Sprite GameGraphic::getLeftPipeSprite () `[inline]`

Getter for pipe sprite.

Returns

pipe sprite.

3.4.3.11 CL_Texture& GameGraphic::getMenuBackground () `[inline]`

Getter for menu background texture.

Returns

background texture.

3.4.3.12 CL_SoundBuffer_Session GameGraphic::getMenuBgMusic () `[inline]`

Getter for menu background music.

Returns

menu background music.

3.4.3.13 std::vector<CL_SoundBuffer_Session> GameGraphic::getMenuFx () `[inline]`

Getter for menu effect sounds.

Returns

menu effect sounds.

3.4.3.14 CL_Sprite GameGraphic::getMovingFabioSprite () `[inline]`

Getter for fabio sprite.

Returns

fabio sprite.

3.4.3.15 CL_Sprite GameGraphic::getMovingGoombaSpriteL () `[inline]`

Getter for goomba sprite.

Returns

goomba.

3.4.3.16 CL_Sprite GameGraphic::getMovingGoombaSpriteR () `[inline]`

Getter for goomba sprite.

Returns

goomba.

3.4.3.17 CL_Sprite GameGraphic::getMovingKoopaSpriteL () `[inline]`

Getter for koopa sprite.

Returns

koopa.

3.4.3.18 CL_Sprite GameGraphic::getMovingKoopaSpriteR () `[inline]`

Getter for koopa.

Returns

koopa.

3.4.3.19 CL_Sprite GameGraphic::getMovingSuperFabioSprite () `[inline]`

Getter for fabio sprite.

Returns

fabio sprite.

3.4.3.20 CL_Sprite GameGraphic::getMovingSuperUoleviSprite () `[inline]`

Getter for uolevi sprite.

Returns

uolevi sprite.

3.4.3.21 CL_Sprite GameGraphic::getMovingTileSprite () `[inline]`

Getter for tile sprite.

Returns

tile.

3.4.3.22 CL_Sprite GameGraphic::getMovingUoleviSprite () `[inline]`

Getter for uolevi sprite.

Returns

uolevi sprite.

3.4.3.23 CL_Sprite GameGraphic::getRightPipeSprite () `[inline]`

Getter for pipe sprite.

Returns

pipe sprite.

3.4.3.24 CL_Sprite GameGraphic::getShroomSprite () `[inline]`

Getter for shroom sprite.

Returns

shroom.

3.4.3.25 CL_Sprite GameGraphic::getStillFabioSprite () `[inline]`

Getter for fabio sprite.

Returns

fabio sprite.

3.4.3.26 CL_Sprite GameGraphic::getStillGoombaSprite () `[inline]`

Getter for goomba sprite.

Returns

goomba.

3.4.3.27 CL_Sprite GameGraphic::getStillKoopaSprite () `[inline]`

Getter for koopa sprite.

Returns

koopa.

3.4.3.28 CL_Sprite GameGraphic::getStillSuperFabioSprite () `[inline]`

Getter for fabio sprite.

Returns

fabio sprite.

3.4.3.29 CL_Sprite GameGraphic::getStillSuperUoleviSprite () `[inline]`

Getter for uolevi sprite.

Returns

uolevi sprite.

3.4.3.30 CL_Sprite GameGraphic::getStillTileSprite () `[inline]`

Getter for koopa sprite.

Returns

koopa.

3.4.3.31 CL_Sprite GameGraphic::getStillUoleviSprite () `[inline]`

Getter for uolevi sprite.

Returns

uolevi sprite.

3.4.3.32 GameGraphic& GameGraphic::operator= (GameGraphic & *other*) `[inline]`

Assignment-operator.

Parameters

other reference to the object to be assigned.

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

- gamegraphic.hh

3.5 GameIO Struct Reference

GameIO struct.

```
#include <gameio.hh>
```

Public Member Functions

- GameIO ()

  Constuctor.

- GameIO (GameIO &gameio)

  Copy constructor.

- ∼GameIO ()

  Destructor.

- GameIO & operator= (GameIO &other)

  Assignment operator.

- CL_GraphicContext ∗ getGCptr ()

  Get GC pointer.

- CL_GraphicContext & getGC ()

  Get GC.

- CL_InputDevice & getKeyboard ()

  Get keyboard.

- CL_InputDevice & getMouse ()

  Get mouse.

- CL_DisplayWindow & getWindow ()

  Get window.

3.5.1 Detailed Description

GameIO struct. GameIO struct is a container for all game IO.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 GameIO::GameIO (GameIO & *gameio*) `[inline]`

Copy constructor.

Parameters

gameio Other gameIO.

3.5.3 Member Function Documentation

3.5.3.1 CL_GraphicContext& GameIO::getGC () `[inline]`

Get GC.

Returns

GC reference.

3.5.3.2 CL_GraphicContext∗ GameIO::getGCptr () `[inline]`

Get GC pointer.

Returns

GC pointer.

3.5.3.3 CL_InputDevice& GameIO::getKeyboard () `[inline]`

Get keyboard.

Returns

Reference to keyboard ID.

3.5.3.4 CL_InputDevice& GameIO::getMouse () `[inline]`

Get mouse.

Returns

Reference to mouse ID.

3.5.3.5 CL_DisplayWindow& GameIO::getWindow () `[inline]`

Get window.

Returns

Reference to the game window.

3.5.3.6 GameIO& GameIO::operator= (GameIO & *other*) `[inline]`

Assignment operator.

Parameters

other Other gameIO.

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

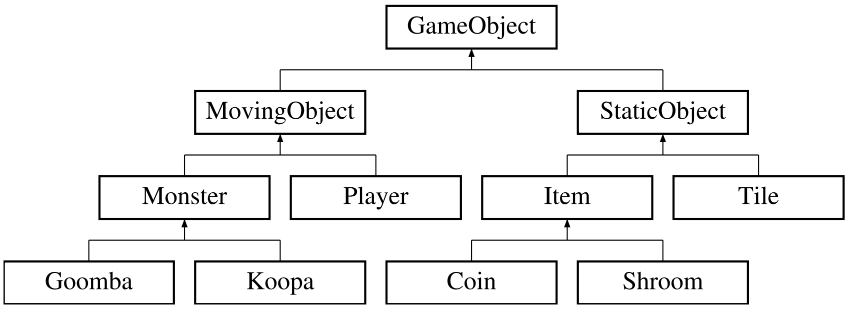
- gameio.hh

3.6 GameObject Class Reference

An abstract class for all functional objects in the game.

`#include <gameobject.hh>`

Inheritance diagram for GameObject:



Public Member Functions

- GameObject (float init_x, float init_y, GameIO &gameIO)

    constructor.

- void drawObject ()

    Draws the object.

- std::pair< float, float > getCoordinates ()

    A pure virtual destructor.

- void setCoordinates (std::pair< float, float > newCoords)

    Set coodinates.

- CL_Sprite getSprite ()

    Get sprite.

- CL_CollisionOutline & getOutline ()

    Gets the collision outline.

- bool collides (GameObject o)

    Collision check for moving objs.

- virtual GameObject & operator= (const GameObject other)

    virtual assignment operator.

Protected Attributes

- std::pair< float, float > coordinates

    Object coordinates.

- CL_Sprite objectSprite

  Object sprite.

- std::pair< int, int > size

  Object size (rectangle).

- CL_CollisionOutline outline

  Collision outline for the object.

- GameIO & gameIO

  game io reference.

3.6.1 Detailed Description

An abstract class for all functional objects in the game. ABSTRACT superclass for all functional objects in the game. e.g players, monsters, items and tiles.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 GameObject::GameObject (float *init_x*, float *init_y*, GameIO & *gameIO*) `[inline]`

constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO reference to gameio struct.

3.6.3 Member Function Documentation

3.6.3.1 bool GameObject::collides (GameObject *o*) `[inline]`

Collision check for moving objs.

Collision is checked by calculating the angle of the impact. It is calculated by checking the relation of speed vector and the impact point.

Parameters

o Moving object that is checked.

Returns

boolean if the collision happened.

3.6.3.2 void GameObject::drawObject () `[inline]`

Draws the object.

Draws the object into the game's graphic context into its own coordinates.

Reimplemented in Tile.

3.6.3.3 std::pair<float,float> GameObject::getCoordinates () `[inline]`

A pure virtual destructor.

Gets the coordinates.

Gets the coordiates.

Returns

The coordinates.

3.6.3.4 CL_CollisionOutline& GameObject::getOutline () `[inline]`

Gets the collision outline.

Returns

collision outline.

3.6.3.5 CL_Sprite GameObject::getSprite () `[inline]`

Get sprite.

Gets the sprite-image of the object.

Returns

The image-file.

3.6.3.6 virtual GameObject& GameObject::operator= (const GameObject *other*) `[inline, virtual]`

virtual assignment operator.

Parameters

other object to be assigned.

Returns

the assigned object.

3.6.3.7 void GameObject::setCoordinates (std::pair< float, float > *newCoords*) `[inline]`

Set coodinates.

Sets the coordiates.

Parameters

newCoords new coordinates.

3.6.4 Field Documentation

3.6.4.1 std::pair<float, float> GameObject::coordinates `[protected]`

Object coordinates.

The coordinates of the object.

3.6.4.2 CL_Sprite GameObject::objectSprite `[protected]`

Object sprite.

The image of the object.

3.6.4.3 std::pair<int, int> GameObject::size `[protected]`

Object size (rectangle).

The size of the sprite int int.

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

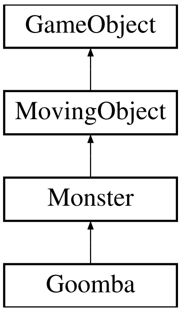
- gameobject.hh

3.7 Goomba Class Reference

A normal class for Goombas.

`#include <goomba.hh>`

Inheritance diagram for Goomba:



Public Member Functions

- Goomba (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef, std::pair< float, float > init_velocity)

  A constructor.

- ∼Goomba ()

  A destructor.

- Goomba (Goomba &other)

  A copy constructor.

- Coin & reward ()

  A reward.

- std::pair< float, float > getNextCoordinates ()

  A get next coords.

- void setVelocity (std::pair< float, float > newVelo)

  Setter for velocity.

- std::pair< float, float > getVelocity ()

  getter for velocity.

- std::pair< float, float > move ()

  moves the monster.

- void hitWall ()

  A pure virtual hitwall.

- void hitFloor ()

  A pure virtual hitbyfloor.

- void hitByPlayer ()

  A pure virtual hitbyplayer.

- Goomba & operator= (Goomba &other)

  Assignment operator.

3.7.1 Detailed Description

A normal class for Goombas. goomba class for easy sample monster. inherits Monster and GameObject classes inherited funtionalities: -drawing -collisions -virtual move() -generation -health

own functionalities: -AI (boolean for which direction it will move) -move() implementation -sprite

3.7.2 Constructor & Destructor Documentation

3.7.2.1 Goomba::Goomba (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*, std::pair< float, float > *init_velocity*)

A constructor.

A normal class for Goombas.

Parameters

init_x initial x coordinate.

init_y initial y coordinate.

gameIO reference to gameIO.

gameGraphicRef reference to gameGraphic.

init_velocity initial velocity. Also loads the image by the image path into the objectSprite.

goomba class for easy sample monster. inherits Monster and GameObject classes inherited funtionalities: -drawing -collisions -virtual move() -generation -health

own functionalities: -AI (boolean for which direction it will move) -move() implementation -sprite A constructor.

Parameters

init_x initial x coordinate.

init_y initial y coordinate.

gameIO reference to gameIO.

gameGraphicRef reference to gameGraphic.

init_velocity initial velocity. Also loads the image by the image path into the objectSprite.

3.7.2.2 Goomba::∼Goomba ()

A destructor.

RO3.

3.7.2.3 Goomba::Goomba (Goomba & *other*)

A copy constructor.

Parameters

other reference to goomba to be copied.

3.7.3 Member Function Documentation

3.7.3.1 std::pair< float, float > Goomba::getNextCoordinates () `[virtual]`

A get next coords.

Returns

next coordinates.

Implements Monster.

3.7.3.2 std::pair< float, float > Goomba::getVelocity () `[virtual]`

getter for velocity.

Returns

the velocity.

Implements Monster.

3.7.3.3 std::pair< float, float > Goomba::move () `[virtual]`

moves the monster.

Returns

new coordinates.

Reimplemented from MovingObject.

3.7.3.4 Goomba & Goomba::operator= (Goomba & *other*)

Assignment operator.

Parameters

other reference to the goomba to be assigned.

Returns

reference to the new goomba.

3.7.3.5 Coin & Goomba::reward () `[virtual]`

A reward.

Returns

reference to a coin.

Implements Monster.

3.7.3.6 void Goomba::setVelocity (std::pair< float, float > *newVelo*) `[virtual]`

Setter for velocity.

Sets goomba's velocity.

Parameters

newVelo new velocity.

Implements Monster.

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

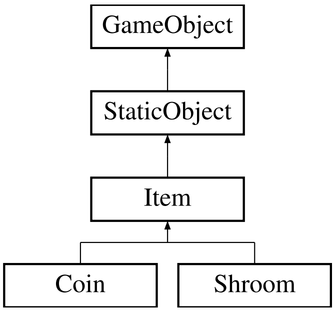
- goomba.hh
- goomba.cc

3.8 Item Class Reference

Abstract class for items.

`#include <item.hh>`

Inheritance diagram for Item:



Public Member Functions

- Item (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef)

    A constructor.

3.8.1 Detailed Description

Abstract class for items. Mainly for making the program more modular.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 Item::Item (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*) `[inline]`

A constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO the reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

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

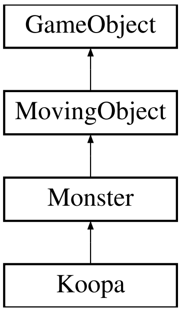
- item.hh

3.9 Koopa Class Reference

A normal class for Koopa Troopas.

```
#include <koopa.hh>
```

Inheritance diagram for Koopa:



Public Member Functions

- Koopa (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef, std::pair< float, float > init_velocity)

  A constructor.

- Koopa (Koopa &)

  A copy constructor.

- ~Koopa ()

  A destructor.

- Coin & reward ()

  A reward.

- std::pair< float, float > getNextCoordinates ()

  A get next coords.

- void setVelocity (std::pair< float, float > newVelo)

  Setter for velocity.

- std::pair< float, float > getVelocity ()

  getter for velocity.

- std::pair< float, float > move ()

  moves the monster.

- void hitWall ()

  A pure virtual hitwall.

- void hitFloor ()

  A pure virtual hitbyfloor.

- void hitByPlayer ()

  A pure virtual hitbyplayer.

- Koopa & operator= (Koopa &other)

  Assignment operator.

3.9.1 Detailed Description

A normal class for Koopa Troopas. koopa class for easy sample monster. inherits Monster and GameObject classes inherited funtionalities: -drawing -collisions -virtual move() -generation -health

own functionalities: -AI (boolean for which direction it will move) -move() implementation -sprite

3.9.2 Constructor & Destructor Documentation

3.9.2.1 Koopa::Koopa (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*, std::pair< float, float > *init_velocity*)

A constructor.

A normal class for Koopa Troopas.

Parameters

init_x initial x coordinate.

init_y initial y coordinate.

gameIO reference to gameIO.

gameGraphicRef reference to gameGraphic.

init_velocity initial velocity. Also loads the image by the image path into the objectSprite.

koopa class for easy sample monster. inherits Monster and GameObject classes inherited funtionalities: -drawing -collisions -virtual move() -generation -health

own functionalities: -AI (boolean for which direction it will move) -move() implementation -sprite A constructor.

Parameters

init_x initial x coordinate.

init_y initial y coordinate.

gameIO reference to gameIO.

gameGraphicRef reference to gameGraphic.

init_velocity initial velocity. Also loads the image by the image path into the objectSprite.

3.9.2.2 Koopa::Koopa (Koopa & *other*)

A copy constructor.

Parameters

other reference to koopa to be copied.

3.9.3 Member Function Documentation

3.9.3.1 std::pair< float, float > Koopa::getNextCoordinates () `[virtual]`

A get next coords.

Returns

next coordinates.

Implements Monster.

3.9.3.2 std::pair< float, float > Koopa::getVelocity () `[virtual]`

getter for velocity.

Returns

the velocity.

Implements Monster.

3.9.3.3 std::pair< float, float > Koopa::move () `[virtual]`

moves the monster.

Returns

new coordinates.

Reimplemented from MovingObject.

3.9.3.4 Koopa & Koopa::operator= (Koopa & *other*)

Assignment operator.

Parameters

other reference to the koopa to be assigned.

Returns

reference to the new koopa.

3.9.3.5 Coin & Koopa::reward () `[virtual]`

A reward.

Creates a coin and returns a reference to it.

Returns

a reference to a coin.

Implements Monster.

3.9.3.6 void Koopa::setVelocity (std::pair< float, float > *newVelo*) `[virtual]`

Setter for velocity.

Sets koopa's velocity.

Parameters

newVelo new velocity.

Implements Monster.

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

- koopa.hh
- koopa.cc

3.10 Menu Class Reference

Menu class.

```
#include <menu.hh>
```

Public Types

- enum MenuSelection { NOSELECTION, CONTINUE, NEW_GAME, QUIT }

  Enumerations for menu selections.

Public Member Functions

- Menu (Options &options, GameIO &gameio, GameGraphic &gameGraphic)

  Constructor.

- ~Menu ()

  Destructor.

- void key (const CL_InputEvent &event, const CL_InputState &state)

  Keyboard input reader.

- MenuSelection drawMain (bool showContinue)

  Draws main menu.

- int getKey ()

  Get key binding.

- void drawOptions ()

3.10.1 Detailed Description

Menu class. Menu class. Drawing and using the menu.

3.10.2 Member Enumeration Documentation

3.10.2.1 enum Menu::MenuSelection

Enumerations for menu selections.

Enumerator:

NOSELECTION No menu selection.

CONTINUE Continue selection from the menu.

NEW_GAME New game selection from the menu.

QUIT Quit selection from the menu.

3.10.3 Constructor & Destructor Documentation

3.10.3.1 Menu::Menu (Options & *options*, GameIO & *gameio*, GameGraphic & *gameGraphic*)

Constructor.

Sets game IO, graphics, sounds, options and signal handling for the menu

Parameters

options Game options.

gameio Game IO.

gameGraphic Game graphics.

3.10.3.2 Menu::∼Menu ()

Destructor.

Placeholder. We may be needing this with the signals because now it segfaults.

3.10.4 Member Function Documentation

3.10.4.1 Menu::MenuSelection Menu::drawMain (bool *showContinue*)

Draws main menu.

Draws main menu.

Parameters

showContinue If game is started, draw menu with a continue button.

Returns

MenuSelection of whether to continue or start a new game or to quit.

3.10.4.2 void Menu::drawOptions ()

Draws options.

3.10.4.3 int Menu::getKey ()

Get key binding.

Helper function that tells the user to press a button to set the key binding.

Returns

The code of the key pressed.

3.10.4.4 void Menu::key (const CL_InputEvent & *event*, const CL_InputState & *state*)

Keyboard input reader.

Reads keyboard signals. Sets the variable "input" to the key pressed. Used to set key bindings.

Parameters

event Tells which key is pressed.

state Reads keyboard signals. Sets the variable "input" to the key pressed. Used to set key bindings.

event Tells which key is pressed.

state State.

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

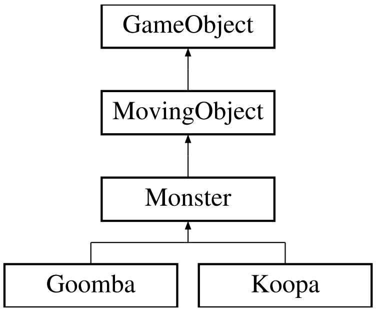
- menu.hh
- menu.cc

3.11 Monster Class Reference

An abstract class for monsters.

`#include <monster.hh>`

Inheritance diagram for Monster:



Public Member Functions

- Monster (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef, std::pair< float, float > init_velocity)

  A constructor.

- virtual Coin & reward ()=0

  A pure virtual destructor.

- virtual std::pair< float, float > getNextCoordinates ()=0

  A pure virtual get next coords.

- virtual void setVelocity (std::pair< float, float > newVelo)=0

  Virtual setter for velocity.

- virtual std::pair< float, float > getVelocity ()=0

  Virtual getter for velocity.

- CL_CollisionOutline & getWinOutline ()

  Getter for outline that kills the monster.

- CL_CollisionOutline & getLoseOutline ()

  Getter for outline that kills the player.

- virtual void hitWall ()=0

  A pure virtual move.

- virtual void hitFloor ()=0

  A pure virtual hitbyfloor.

- virtual void hitByPlayer ()=0

  A pure virtual hitbyplayer.

- bool getIsDead ()

  getter for is dead.

- void kill ()

  function to kill the monster.

Protected Attributes

- bool isDead

  Is the monster dead or alive.

- CL_CollisionOutline winOutline

  Outline that kills the monster.

- CL_CollisionOutline loseOutline

  Outline that kills the player.

3.11.1 Detailed Description

An abstract class for monsters. abstract super class for all the monsters

3.11.2 Constructor & Destructor Documentation

3.11.2.1 Monster::Monster (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*, std::pair< float, float > *init_velocity*) `[inline]`

A constructor.

Parameters

init_x initial x coordinate.

init_y initial y coordinate.

gameIO reference to gameIO.

gameGraphicRef reference to gameGraphic.

init_velocity initial velocity.

3.11.3 Member Function Documentation

3.11.3.1 bool Monster::getIsDead () `[inline]`

getter for is dead.

Returns

isDead.

3.11.3.2 CL_CollisionOutline& Monster::getLoseOutline () `[inline]`

Getter for outline that kills the player.

Returns

the outline.

3.11.3.3 virtual std::pair<float, float> Monster::getVelocity () `[pure virtual]`

Virtual getter for velocity.

Returns

velocity.

Reimplemented from MovingObject.

Implemented in Goomba, and Koopa.

3.11.3.4 CL_CollisionOutline& Monster::getWinOutline () `[inline]`

Getter for outline that kills the monster.

Returns

the outline.

3.11.3.5 virtual void Monster::hitWall () `[pure virtual]`

A pure virtual move.

A pure virtual hitwall.

Implemented in Goomba, and Koopa.

3.11.3.6 virtual Coin& Monster::reward () `[pure virtual]`

A pure virtual destructor.

A pure virtual reward.

Implemented in Goomba, and Koopa.

3.11.3.7 virtual void Monster::setVelocity (std::pair< float, float > *newVelo*) `[pure virtual]`

Virtual setter for velocity.

Parameters

newVelo new velocity.

Reimplemented from MovingObject.

Implemented in Goomba, and Koopa.

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

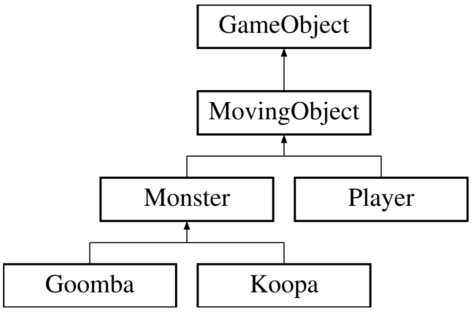
- monster.hh

3.12 MovingObject Class Reference

An abstract class for all moving objects.

`#include <movingobject.hh>`

Inheritance diagram for MovingObject:



Public Member Functions

- MovingObject (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef, std::pair< float, float > init_velocity)

  A constructor.

- virtual std::pair< float, float > move ()

  A pure virtual destructor.

- virtual void setVelocity (std::pair< float, float > newVelo)

  Virtual setter for velocity.

- virtual std::pair< float, float > getVelocity ()

  Virtual getter for velocity.

- bool getInAir ()

  Is object falling.

- void setInAir (bool newAirStatus)

  set object falling status.

Protected Attributes

- std::pair< float, float > velocity

  Object velocity.

- bool inAir

  objects in air status.

- GameGraphic & gameGraphicRef

  Game graphic struct.

3.12.1 Detailed Description

An abstract class for all moving objects. An abstract class for moving GameObjects. Includes pure virtual function Move().

3.12.2 Constructor & Destructor Documentation

3.12.2.1 MovingObject::MovingObject (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*, std::pair< float, float > *init_velocity*) `[inline]`

A constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO the reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

init_velocity initial velocity.

3.12.3 Member Function Documentation

3.12.3.1 bool MovingObject::getInAir () `[inline]`

Is object falling.

Returns

in air status.

3.12.3.2 virtual std::pair<float, float> MovingObject::getVelocity () `[inline, virtual]`

Virtual getter for velocity.

Returns

velocity.

Reimplemented in Goomba, Koopa, Monster, and Player.

3.12.3.3 virtual std::pair<float, float> MovingObject::move () `[inline, virtual]`

A pure virtual destructor.

A virtual move.

Returns

the new coordinates.

Reimplemented in Goomba, Koopa, and Player.

3.12.3.4 void MovingObject::setInAir (bool *newAirStatus*) `[inline]`

set object falling status.

Parameters

newAirStatus new in air status.

3.12.3.5 virtual void MovingObject::setVelocity (std::pair< float, float > *newVelo*) `[inline, virtual]`

Virtual setter for velocity.

Parameters

newVelo new velocity.

Reimplemented in Goomba, Koopa, Monster, and Player.

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

- movingobject.hh

3.13 Options Class Reference

Menu class.

```
#include <options.hh>
```

Public Member Functions

- int getP1KeyLeft ()

  Get P1 key left.

- void setP1KeyLeft (int key)

  Set P1 key left.

- int getP1KeyRight ()

  Get P1 key right.

- void setP1KeyRight (int key)

  Set P1 key right.

- int getP1KeyUp ()

  Get P1 key up.

- void setP1KeyUp (int key)

  Set P1 key up.

- int getP2KeyLeft ()

  Get P2 key left.

- void setP2KeyLeft (int key)

  Set P2 key left.

- int getP2KeyRight ()

  Get P2 key right.

- void setP2KeyRight (int key)

  Set P2 key right.

- int getP2KeyUp ()

  Get P2 key up.

- void setP2KeyUp (int key)

  Set P2 key up.

3.13.1 Detailed Description

Menu class. Data container class for game options.

3.13.2 Member Function Documentation

3.13.2.1 int Options::getP1KeyLeft () `[inline]`

Get P1 key left.

Returns

P1 key left.

3.13.2.2 int Options::getP1KeyRight () `[inline]`

Get P1 key right.

Returns

P1 key right.

3.13.2.3 int Options::getP1KeyUp () `[inline]`

Get P1 key up.

Returns

P1 key up.

3.13.2.4 int Options::getP2KeyLeft () `[inline]`

Get P2 key left.

Returns

P2 key left.

3.13.2.5 int Options::getP2KeyRight () `[inline]`

Get P2 key right.

Returns

P2 key right.

3.13.2.6 int Options::getP2KeyUp () `[inline]`

Get P2 key up.

Returns

P2 key up.

3.13.2.7 void Options::setP1KeyLeft (int *key*) `[inline]`

Set P1 key left.

Parameters

key P1 key left.

3.13.2.8 void Options::setP1KeyRight (int *key*) `[inline]`

Set P1 key right.

Parameters

key P1 key right.

3.13.2.9 void Options::setP1KeyUp (int *key*) `[inline]`

Set P1 key up.

Parameters

key P1 key up.

3.13.2.10 void Options::setP2KeyLeft (int *key*) `[inline]`

Set P2 key left.

Parameters

key P2 key left.

3.13.2.11 void Options::setP2KeyRight (int *key*) `[inline]`

Set P2 key right.

Parameters

key P2 key right.

3.13.2.12 void Options::setP2KeyUp (int *key*) `[inline]`

Set P2 key up.

Parameters

key P2 key up.

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

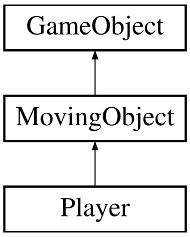
- options.hh

3.14 Player Class Reference

A normal class for players.

```
#include <player.hh>
```

Inheritance diagram for Player:



Public Member Functions

- Player (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef, std::pair< float, float > init_velocity, std::string normalSpritePath, std::string superSpritePath, bool isFabio)

  Constructor.

- Player (const Player &other)

  Copy-constructor.

- ∼Player ()

  A destructor.

- std::pair< float, float > move ()

  Moves the player.

- void facingLeft (bool val)

  Which way the player is facing.

- void setVelocity (std::pair< float, float > newVelo)

  Sets new velocity for player.

- std::pair< float, float > getVelocity ()

  Getter for the velocity of the player.

- void growSuper ()

  Grow into Super Fabio.

- void growNormal ()

  Decrease into normal Fabio.

- void addCoin ()

  Get one more coin.

- int getCoins ()

getter for coin number.

- void hitByMonster ()

  Player is hit by monster.

- bool getIsDead ()

  Getter for death-status.

- bool getIsFabio ()

  Getter for is the player fabio or not.

- float getStarTime ()

  Getter for starTime.

- void setStarTime (float newTime)

  Setter for starTime.

- Player & operator= (const Player &orig)

  assignment operator.

3.14.1 Detailed Description

A normal class for players. class for players: inherits GameObject abstract class so it includes: -drawing -collisions -virtual move() function

functionality for class: move() implementation health (big/small fabio) bonuses

have-to-know info: -own place -own speed -own sprite -own bonuses -own health state

3.14.2 Constructor & Destructor Documentation

3.14.2.1 Player::Player (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*, std::pair< float, float > *init_velocity*, std::string *normalSpritePath*, std::string *superSpritePath*, bool *isFabio*)

Constructor.

A normal class for players.

Constructor. Also initializes the isSuper, isDead and coins variables.

Parameters

init_x Initial x.

init_y Initial y.

gameIO GameIO rerefence.

gameGraphicRef GameGraphic rerefence.

init_velocity Initial Velocity.

normalSpritePath Sprite path for normal fabio.

superSpritePath Sprite path for super fabio.

isFabio Is this player Fabio or not.

class for players: inherits GameObject abstract class so it includes: -drawing -collisions -virtual move() function

functionality for class: move() implementation health (big/small fabio) bonuses

have-to-know info: -own place -own speed -own sprite -own bonuses -own health state Constructor.

Constructor. Also initializes the isSuper, isDead and coins variables.

Parameters

init_x Initial x.

init_y Initial y.

gameIO GameIO rerefence.

gameGraphicRef GameGraphic rerefence.

init_velocity Initial Velocity.

normalSpritePath Sprite path for normal fabio.

superSpritePath Sprite path for super fabio.

isFabio Is this player Fabio or not.

3.14.2.2 Player::Player (const Player & *other*)

Copy-constructor.

Parameters

other Player to copy.

other reference to player-object to be copied.

3.14.2.3 Player::∼Player () `[inline]`

A destructor.

RO3.

3.14.3 Member Function Documentation

3.14.3.1 void Player::addCoin ()

Get one more coin.

gain one more coin.

3.14.3.2 void Player::facingLeft (bool *val*)

Which way the player is facing.

Parameters

val true = left.

3.14.3.3 int Player::getCoins ()

getter for coin number.

get the number of coins.

Returns

INT the number of coins that player has.

3.14.3.4 bool Player::getIsDead ()

Getter for death-status.

Returns

isDead.

3.14.3.5 bool Player::getIsFabio ()

Getter for is the player fabio or not.

Returns

isfabio
isfabio.

3.14.3.6 float Player::getStarTime ()

Getter for starTime.

Returns

starTime.

3.14.3.7 std::pair< float, float > Player::getVelocity () `[virtual]`

Getter for the velocity of the player.

Returns

Velocity of the player.

Reimplemented from MovingObject.

3.14.3.8 std::pair< float, float > Player::move () `[virtual]`

Moves the player.

Moves the player.

Returns

new coordinates.

Reimplemented from MovingObject.

3.14.3.9 Player & Player::operator= (const Player & *orig*)

assignment operator.

Parameters

orig Reference to player-object to be assigned.

Returns

this.

3.14.3.10 void Player::setStarTime (float *newTime*)

Setter for starTime.

Parameters

newTime new startime.

3.14.3.11 void Player::setVelocity (std::pair< float, float > *newVelo*) `[virtual]`

Sets new velocity for player.

Sets new velocity.

Parameters

newVelo new velocity.

Reimplemented from MovingObject.

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

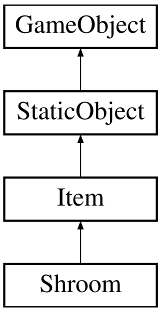
- player.hh
- player.cc

3.15 Shroom Class Reference

A header for shrooms.

`#include <shroom.hh>`

Inheritance diagram for Shroom:



Public Member Functions

- Shroom (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef)

  A constructor.

- Shroom (const Shroom &other)

  Copy-constructor.

- Shroom & operator= (const Shroom &other)

  assignment operator.

- void gather ()

  Gathers the mushroom.

- ∼Shroom ()

  A destructor.

3.15.1 Detailed Description

A header for shrooms.

3.15.2 Constructor & Destructor Documentation

3.15.2.1 Shroom::Shroom (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*)

A constructor.

A normal class for super mushrooms.

Parameters

init_x initial x.

init_y initial y.

gameIO reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

shroom class is example of item subclasses

inherits abstract Item and GameObject classes inherited functionalities: -drawing -collisions -virtual move() -generation

own functionalities: -move() -grant bonus A constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO reference to gameio struct.

gameGraphicRef reference to gamegraphic struct.

3.15.2.2 Shroom::Shroom (const Shroom & *other*)

Copy-constructor.

Parameters

other reference to the object to be copied.

3.15.2.3 Shroom::∼Shroom ()

A destructor.

RO3.

3.15.3 Member Function Documentation

3.15.3.1 Shroom & Shroom::operator= (const Shroom & *other*)

assignment operator.

Parameters

other reference to player-object to be assigned.

Returns

this.

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

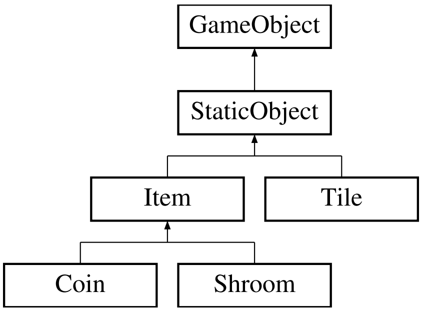
- shroom.hh
- shroom.cc

3.16 StaticObject Class Reference

An abstract class for all non-moving objects.

`#include <staticobject.hh>`

Inheritance diagram for StaticObject:



Public Member Functions

- StaticObject (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef)

  A constructor.

- StaticObject (const StaticObject &other)

  Copy-constructor.

- StaticObject & operator= (const StaticObject &other)

  assignment operator.

Protected Attributes

- GameGraphic & gameGraphicRef

  a reference to gamegraphic object.

3.16.1 Detailed Description

An abstract class for all non-moving objects. Abstract class for non-moving gameobjects. Mainyl for making the program more modular.

3.16.2 Constructor & Destructor Documentation

3.16.2.1 StaticObject::StaticObject (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*) `[inline]`

A constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO the reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

3.16.2.2 StaticObject::StaticObject (const StaticObject & *other*) `[inline]`

Copy-constructor.

Parameters

other reference to the object to be copied.

3.16.3 Member Function Documentation

3.16.3.1 StaticObject& StaticObject::operator= (const StaticObject & *other*) `[inline]`

assignment operator.

Parameters

other reference to player-object to be assigned.

Returns

this.

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

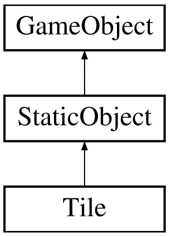
- staticobject.hh

3.17 Tile Class Reference

Class for a single tile object.

`#include <tile.hh>`

Inheritance diagram for Tile:



Public Member Functions

- Tile (float init_x, float init_y, GameIO &gameIO, GameGraphic &gameGraphicRef)

  A constructor.

- Tile (const Tile &other)

  Copy-constructor.

- ∼Tile ()

  A destructor.

- void hitByPlayer ()

  Function for player hitting the tile.

- void drawObject ()

  Draws the object.

- void returnToInitPos ()

  Returns the tile to initial position.

- Tile & operator= (const Tile &other)

  assignment operator.

3.17.1 Detailed Description

Class for a single tile object.

3.17.2 Constructor & Destructor Documentation

3.17.2.1 Tile::Tile (float *init_x*, float *init_y*, GameIO & *gameIO*, GameGraphic & *gameGraphicRef*)

A constructor.

A normal class for tiles.

Parameters

init_x initial x.

init_y initial y.

gameIO the reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

tile class is for tiles and maybe later on for moving tiles or POW-tiles

inherits the GameObject class inherited functionalities: -drawing -collisions -virtual move()

own functionalities: -sprite -move() when collided with A constructor.

Parameters

init_x initial x.

init_y initial y.

gameIO the reference to gameIO struct.

gameGraphicRef reference to gamegraphic struct.

3.17.2.2 Tile::Tile (const Tile & *other*)

Copy-constructor.

Parameters

other reference to the object to be copied.

3.17.3 Member Function Documentation

3.17.3.1 void Tile::drawObject ()

Draws the object.

Draws the object into the game's graphic context into its own coordinates.

Reimplemented from GameObject.

3.17.3.2 void Tile::hitByPlayer ()

Function for player hitting the tile.

Animates the tile.

3.17.3.3 Tile & Tile::operator= (const Tile & *other*)

assignment operator.

Parameters

other reference to player-object to be assigned.

Returns

this.

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

- tile.hh
- tile.cc

Index