Genetic Tetris

Generated by Doxygen 1.9.0

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Namespace Documentation

4.1 genetic_tetris Namespace Reference

Classes

- class AI
- class App
- class Button
- class Controller
- class EventManager
- class EvolutionaryAlgo
- class EvolveController
- class GameController
- class Genome
- class GenomeFileNotFoundException
- class GUI
- class IncDecDialog
- class MenuController
- class Move
- class ObservableTetris
- class Observer
- class RandomNumberGenerator
- class SoundManager
- class Subject
- class Tetris
- class TetrisBoard
- class Tetromino
- class TetrominoGenerator

Enumerations

- enum EventType {
  TETROMINO_DROPPED, **PLAY_BUTTON_CLICKED**, **START_GAME_BUTTON_CLICKED**, **EVOLVE_↩ BUTTON_CLICKED**,
  EXIT_BUTTON_CLICKED, **BACK_BUTTON_CLICKED**, **SAVE_BUTTON_CLICKED**, **START_EVOLVE_↩ BUTTON_CLICKED**,
  GENOMES_SAVED, **GENERATION_OUT_OF_BOUNDS**, **GAME_STARTED**, **GAME_START_FAILED** }

Functions

- const std::map< Tetromino::Color, sf::Color > & **getTetrominoColorMap** ()

4.1.1 Detailed Description

This file contains helper classes used in GUI.

4.1.2 Enumeration Type Documentation

4.1.2.1 EventType

```
enum genetic_tetris::EventType  [strong]
```

Enum specifying possible events in the application

Chapter 5

Class Documentation

5.1 genetic_tetris::AI Class Reference

`#include <ai.hpp>`

Inheritance diagram for genetic_tetris::AI:

Public Member Functions

- **AI** (Tetris &tetris)
- virtual void finish ()

  Tells algorithm to finish.
- virtual void drop ()=0

  Tells algorithm to make a move.
- void resetTetris ()

  Resets genetic_tetris::Tetris state.

Protected Attributes

- Tetris & **tetris_**
- RandomNumberGenerator & **generator_**
- volatile bool **finish_** = false

5.1.1 Detailed Description

Base class for all algorithms working with genetic_tetris::Tetris

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/AI/ai.hpp

5.2 genetic_tetris::App Class Reference

`#include <app.hpp>`

Public Member Functions

- void **run** ()
- void **update** ()
- void **display** ()

Private Types

- enum **State** { **MENU**, **PLAYING**, **EVOLVING**, **CLOSED** }

Private Member Functions

- void **pollSfmlEvents** ()
- void **pollCustomEvents** ()
- void **close** ()
- void **start** ()
- void **reset** ()

Private Attributes

- EventManager & **event_manager_**
- SoundManager & **sound_manager_**
- ObservableTetris **tetris_human_**
- Tetris **tetris_ai_**
- EvolutionaryAlgo **ai_**
- GUI **gui_**
- GameController **game_controller_**
- EvolveController **evolve_controller_**
- MenuController **menu_controller_**
- State **state_**
- Controller * **active_controller_**

Static Private Attributes

- static const int **WINDOW_WIDTH_** = 800
- static const int **WINDOW_HEIGHT_** = 900
- static const int **FPS_** = 60

5.2.1 Detailed Description

Main class managing the application. It aggregates all the data and controls current state. Logic and displaying information is pushed to corresponding controllers.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/app.hpp

5.3 genetic_tetris::Button Class Reference

`#include <gui_utils.hpp>`

Inheritance diagram for genetic_tetris::Button:

Public Member Functions

- void **setPosition** (const sf::Vector2f &pos)
- void **setSize** (const sf::Vector2f &size)
- void **setText** (const std::string &text, const sf::Font &font, int size=24)
- void **update** ()
- void **handleEvent** (const sf::Event &e, const sf::Window &window)
- void **setOnClick** (std::function< void()> on_click)

Protected Member Functions

- void **draw** (sf::RenderTarget &target, sf::RenderStates states) const override

Private Types

- enum **State** { **NORMAL**, **CLICKED** }

Private Attributes

- const sf::Time **CLICK_ANIMATION_TIME_** = sf::seconds(0.1f)
- const sf::Color **CLICK_HUE_CHANGE_** = sf::Color(20, 20, 20, 0)
- enum genetic_tetris::Button::State **state_** = State::NORMAL
- sf::Color **text_color_** = sf::Color::White
- sf::Color **bg_color_** = sf::Color(0x708090ff)
- sf::Text **text_**
- sf::Font **font_**
- sf::RectangleShape **rect_**
- SoundManager & **sound_manager_**
- sf::Clock **clock_**
- std::function< void()> **on_click_**

5.3.1 Detailed Description

Custom button class. Provides with only most basic functionalities like custom click handlers, hue change on clicked.

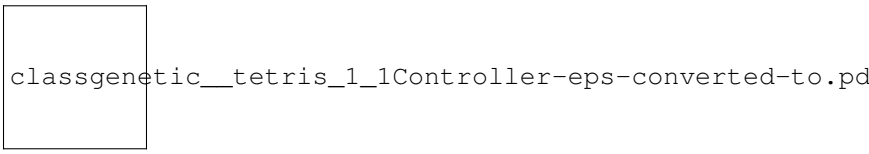
The documentation for this class was generated from the following file:

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/gui/gui_utils.hpp

5.4 genetic_tetris::Controller Class Reference

`#include <controller.hpp>`

Inheritance diagram for genetic_tetris::Controller:



Public Member Functions

- **Controller** (GUI &gui)
- virtual void **update** ()=0
- virtual void **start** ()=0
- virtual void **reset** ()=0
- virtual void **finish** ()=0
- virtual void **handleSfmlEvent** (const sf::Event &e)=0
- virtual void **handleCustomEvent** (EventType e)=0

Protected Attributes

- GUI & **gui_**

5.4.1 Detailed Description

Interface for controller classes. Controllers are used to split logic into more manageable parts.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/controller/controller.hpp

5.5 genetic_tetris::EventManager Class Reference

`#include <event_manager.hpp>`

Public Member Functions

- **EventManager** (const EventManager &)=delete
- EventManager **operator=** (const EventManager &)=delete
- EventType pollEvent ()

  Returns event from the front and removes it.
- void **addEvent** (const EventType &e)
- bool **isEmpty** () const
- void **removeEvent** (EventType event)

Static Public Member Functions

- static EventManager & **getInstance** ()

Private Attributes

- std::list< EventType > **events**

5.5.1 Detailed Description

Simple event manager.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/event_manager.hpp

5.6 genetic_tetris::EvolutionaryAlgo Class Reference

`#include <evolutionary_algo.hpp>`

Inheritance diagram for genetic_tetris::EvolutionaryAlgo:

Public Types

- enum Mode { **PLAY**, **EVOLVE** }

  Specifies mode in which to run the algorithm.

Public Member Functions

- **EvolutionaryAlgo** (Tetris &tetris)
- void operator() (Mode mode)
- void drop () override

  Tells algorithm to make a move.
- void **update** (EventType e) override
- void finish () override

  Tells algorithm to finish.
- void tick ()

  Performs genetic_tetris::Tetris::tick()
- bool isDroppingSmoothly () const

  TODO.
- std::string getInfo () const
- Genome getBest () const

  Returns current best genome.
- void save ()

  Saves genomes to file.
- void setPlayingGeneration (int value)

  Specifies generation number used to play against the player.
- int getAvailableGenerations () const

  Returns the number of generations available in genome file.
- bool getSuccess () const

Static Public Member Functions

- static Move generateBestMove (const Genome &genome, Tetris &tetris)

Private Types

- enum State { **STOP**, **START** }

  Specifies available states of the algorithm execution.

Private Member Functions

- void play ()

  Runs algorithm in mode playing with the player.
- void evolve ()

  Runs algorithm in evolving mode.
- std::vector< Genome > nextGeneration (std::vector< Genome > &pop)

  Generates next generation.
- std::vector< Genome > initialPop ()

  Creates initial population.
- std::vector< Genome > selection (std::vector< Genome > &pop)

  Performs selection.
- std::vector< Genome > mutation (std::vector< Genome > &selected)

  Performs mutation on selected.
- void evaluation (std::vector< Genome > &next_pop)

  Evaluates the next population.
- void mutate (Genome &genome)

  Mutates one genome.

Static Private Member Functions

- static void saveToJSON (const std::string &file, std::vector< Genome > &genomes)

  Saves given set of genomes to specified file.
- static std::vector< Genome > loadFromJSON (const std::string &file)

  Loads set of genomes from specified file.

Private Attributes

- const std::size_t POP_SIZE = 50

  Population size.
- const float MUTATION_RATE = 0.1f

  Rate at which genome attributes will be mutated.
- const float MUTATION_STEP = 0.2f

  Strength of the singular mutation.
- const int MOVES_TO_SIMULATE = 400

  Number of moves simulated in evaluation function.
- const std::string GENOMES_FILE = "res/genomes.json"

  File where genomes will be located.
- State state_ = State::STOP

  Current execution state.
- bool success_

  Execution status.
- Genome best_

  Current best genome.

- std::vector< Genome > generation_bests_

  Best genomes from each generations (.
- float mean_fitness_ = 0.0f

  Mean fitness of last generation.
- int t_ = 0

  Generation count.
- std::mutex m_

  Mutex used to manage std::condition_variable.
- std::condition_variable drop_cond_

  Blocks algorithm if there is nothing for it to be done.
- bool drop_

  If true tells algorithm to drop current tetromino.
- bool smooth_drop_

  Tells whether tetromino should be dropped smoothly.
- bool is_dropping_smoothly_

  Tells whether algorithm is in the process of smoothly dropping a tetromino.
- int playing_generation_

  Generation playing againt the player. Specified in GUI.
- int available_generations_

  Generations available in loaded JSON file containing genomes.

Additional Inherited Members

5.6.1 Detailed Description

Evolutionary algorithm implementation being able to play Tetris

Algorithm can be run in two modes:

- PLAY load population from file and use specified genome to generate moves
- EVOLVE evolve population to create better genomes

5.6.2 Member Function Documentation

5.6.2.1 generateBestMove()

```
static Move genetic_tetris::EvolutionaryAlgo::generateBestMove (
            const Genome & genome,
            Tetris & tetris )  [static]
```

Generates best possible move taking into account tetris state and genome attributes

Parameters

genome	- genome used to calculate fitness function
tetris	- tetris object for which function will generate the best move

Returns

best move generated for current state of tetris

5.6.2.2 getInfo()

```
std::string genetic_tetris::EvolutionaryAlgo::getInfo ( ) const
```

Returns attributes of current generation of algorithm

Returns

string containing information like mean fitness, best score, best genome attributes

5.6.2.3 getSuccess()

```
bool genetic_tetris::EvolutionaryAlgo::getSuccess ( ) const  [inline]
```

Returns algorithm status for play() or evolve()

Returns

true if everything was ok, false e.g. number of available generations was less than playing generation

5.6.2.4 operator()()

```
void genetic_tetris::EvolutionaryAlgo::operator() (
            Mode mode )
```

Runs the algorithm

Parameters

mode	mode in which algorithm will be run

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/AI/evolutionary_algo.hpp

5.7 genetic_tetris::EvolveController Class Reference

```
#include <evolve_controller.hpp>
```

Inheritance diagram for genetic_tetris::EvolveController:

Public Types

- enum **State** { **START**, **STOP** }

Public Member Functions

- **EvolveController** (Tetris &tetris_ai, EvolutionaryAlgo &ai, GUI &gui)
- void **update** () override
- void **start** () override
- void **reset** () override
- void **finish** () override
- void **handleSfmlEvent** (const sf::Event &) override
- void **handleCustomEvent** (EventType e) override

Public Attributes

- enum genetic_tetris::EvolveController::State **state_** = State::STOP

Private Attributes

- Tetris & **tetris_ai_**
- EvolutionaryAlgo & **ai_**
- std::thread **ai_thread_**

Additional Inherited Members

5.7.1 Detailed Description

Evolve screen controller

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/controller/evolve_controller.hpp

5.8 genetic_tetris::GameController Class Reference

`#include <game_controller.hpp>`

Inheritance diagram for genetic_tetris::GameController:

Public Types

- enum **State** { **START**, **STOP** }

Public Member Functions

- **GameController** (ObservableTetris &tetris_human, EvolutionaryAlgo &ai, GUI &gui)
- void **update** (EventType e) override
- void **update** () override
- void **start** () override
- void **reset** () override
- void **finish** () override
- void **handleSfmlEvent** (const sf::Event &event) override
- void **handleCustomEvent** (EventType e) override

Public Attributes

- enum genetic_tetris::GameController::State **state_** = State::STOP

Private Member Functions

- void **humanTick** (bool is_soft_drop=false)
- void **handlePlayerInput** (const sf::Event &event)

Private Attributes

- const sf::Time AI_MOVE_INTERVAL_ = sf::seconds(0.1f)

  Interval between next AI moves (when player has finished)
- const sf::Time **HARD_DROP_LOCK_DELAY_** = sf::seconds(0.25f)
- const float **DEFAULT_SOFT_DROP_INTERVAL_** = 0.05f
- ObservableTetris & **tetris_human_**
- EvolutionaryAlgo & **ai_**
- sf::Clock **ai_clock_**
- sf::Clock **game_clock_**
- sf::Time **tick_interval_**
- sf::Time **soft_drop_interval_**
- std::thread **ai_thread_**
- SoundManager & **sound_manager_**
- bool **hard_drop_lock_**

Additional Inherited Members

5.8.1 Detailed Description

Game screen controller

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/controller/game_controller.hpp

5.9 genetic_tetris::Genome Class Reference

`#include <genome.hpp>`

Public Member Functions

- Genome ()

  Constructs genome with randomly generated attributes.

Public Attributes

- long id

  Genome id.
- float rows_cleared

  Weight for rows cleared in the last move.
- float max_height

  Weight for maximum column height.
- float cumulative_height

  Weight for sum of heights of all columns.
- float relative_height

  Difference between highest and lowest column.
- float holes

  Sum of holes.
- float roughness

  Sum of height differences of all adjacent columns.
- float score

  Last genome score.

Static Public Attributes

- static long next_id = 0

  Next genome id.

5.9.1 Detailed Description

Genome used by evolutionary algorithm. It contains several attributes being the weights algorithm's fitness function. Fitness_function is a sum of attribute_weight ∗ attribute_value for all attributes.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/AI/genome.hpp

5.10 genetic_tetris::GenomeFileNotFoundException Class Reference

Inheritance diagram for genetic_tetris::GenomeFileNotFoundException:

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/exception.hpp

5.11 genetic_tetris::GUI Class Reference

`#include <gui.hpp>`

Public Types

- enum **ScreenType** { **MENU**, **GAME**, **EVOLVE** }

Public Member Functions

- **GUI** (int width, int height, int fps, Tetris &human_tetris, Tetris &ai_tetris, EvolutionaryAlgo &ai)
- void **update** ()
- void **draw** ()
- void **close** ()
- bool **pollEvent** (sf::Event &event)
- void **handleSfmlEvent** (const sf::Event &event)
- void **handleCustomEvent** (EventType event)
- void **reset** ()
- void **setActiveScreen** (ScreenType screen_type)
- Screen ∗ **getActiveScreen** ()

Private Attributes

- sf::RenderWindow **window_**
- MenuScreen **menu_screen_**
- GameScreen **game_screen_**
- EvolveScreen **evolve_screen_**
- Screen * **active_screen_**

5.11.1 Detailed Description

Manages screen switching. Displaying is handled by Screen classes.

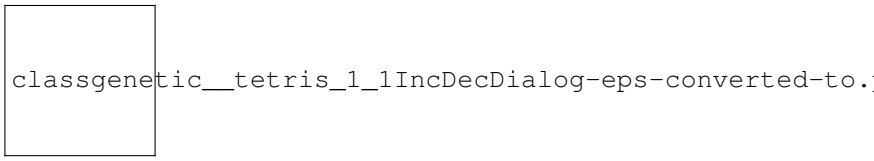
The documentation for this class was generated from the following file:

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/gui/gui.hpp

5.12 genetic_tetris::IncDecDialog Class Reference

`#include <gui_utils.hpp>`

Inheritance diagram for genetic_tetris::IncDecDialog:



Public Member Functions

- IncDecDialog & **setPosition** (const sf::Vector2f &pos)
- IncDecDialog & **setFont** (const sf::Font &font, int size=24)
- IncDecDialog & **setValueBounds** (const sf::Vector2i &bounds)
- void **build** ()
- void **update** ()
- void **handleEvent** (const sf::Event &e, const sf::Window &window)
- int **getValue** () const
- void **setValue** (int value)

Protected Member Functions

- void **draw** (sf::RenderTarget &target, sf::RenderStates states) const override

Private Attributes

- const sf::Vector2f **PLUS_BUTTON_RELATIVE_POS_** = sf::Vector2f(0, -20)
- const sf::Vector2f **MINUS_BUTTON_RELATIVE_POS_** = sf::Vector2f(0, 24)
- const sf::Vector2f **VALUE_TEXT_RELATIVE_POS_** = sf::Vector2f(0, 0)
- const sf::Vector2f **BUTTONS_DEFAULT_SIZE_** = sf::Vector2f(20, 20)
- const int **FONT_DEFAULT_SIZE_** = 24
- sf::Vector2f **dialog_pos_**
- sf::Vector2i **value_bounds_**
- Button **plus_button_**
- Button **minus_button_**
- sf::Vector2f **button_size_**
- sf::Text **value_text_**
- int **value_**
- sf::Font **font_**
- int **font_size_**

5.12.1 Detailed Description

Counter dialog consisting of plus and minus buttons and counter value.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/gui/gui_utils.hpp

5.13 genetic_tetris::MenuController Class Reference

`#include <menu_controller.hpp>`

Inheritance diagram for genetic_tetris::MenuController:

Public Member Functions

- **MenuController** (GUI &gui)
- void **update** () override
- void **start** () override
- void **reset** () override
- void **finish** () override
- void **handleSfmlEvent** (const sf::Event &) override
- void **handleCustomEvent** (EventType) override

Additional Inherited Members

5.13.1 Detailed Description

Menu screen controller. It is a dummy class used so we don't have to make additional checks for active_controller_ pointer

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/controller/menu_controller.hpp

5.14 genetic_tetris::Move Class Reference

Public Member Functions

- **Move** (int moveX, int rotations)
- **Move** (const Move &other)
- Move & **operator=** (const Move &other)
- void apply (Tetris &tetris, bool hard_drop=true)
- int **getMoveX** () const
- int **getRotation** () const
- int **getMaxHeight** () const
- int **getCumulativeHeight** () const
- int **getRelativeHeight** () const
- int **getHoles** () const
- int **getRoughness** () const

Static Public Attributes

- static const int **MIN_MOVE** = -1
- static const int **MAX_MOVE** = Tetris::GRID_WIDTH - 1
- static const int **MIN_ROT** = 0
- static const int **MAX_ROT** = 3

Private Member Functions

- void calculateGridProperties (const Tetris &tetris)

  Calculates grid properties after Move::apply.

Static Private Member Functions

- static int **calculateHoles** (const Tetris::Grid &grid)

Private Attributes

- int move_x_

  Move in x direction.
- int rotations_

  Number of rotations.
- int max_height_ = 0

  Maximum height.
- int cumulative_height_ = 0

  Sum of all column heights.
- int relative_height_ = 0

  Difference between lowest and highest column.
- int holes_ = 0

  Number of holes in a grid.
- int roughness_ = 0

  Sum of height differences of all adjacent columns.

5.14.1 Member Function Documentation

5.14.1.1 apply()

```
void genetic_tetris::Move::apply (
            Tetris & tetris,
            bool hard_drop = true )
```

Performs the move

Parameters

tetris	tetris on which move will be applied
hard_drop	if true will perform hard drop and calculate tetris properties

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/AI/move.hpp

5.15 genetic_tetris::ObservableTetris Class Reference

Inheritance diagram for genetic_tetris::ObservableTetris:

Private Member Functions

- void **generateTetromino** () override

Additional Inherited Members

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/tetris/tetris.hpp

5.16 genetic_tetris::Observer Class Reference

`#include <utils.hpp>`

Inheritance diagram for genetic_tetris::Observer:

Public Member Functions

- virtual void **update** (EventType e)=0

5.16.1 Detailed Description

Abstract observer

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/utils.hpp

5.17 genetic_tetris::RandomNumberGenerator Class Reference

`#include <random_number_generator.hpp>`

Public Member Functions

- **RandomNumberGenerator** (const RandomNumberGenerator &)=delete
- RandomNumberGenerator & **operator=** (const RandomNumberGenerator &)=delete
- float **random_0_1** ()
- template<int a, int b>
  float random ()

Static Public Member Functions

- static RandomNumberGenerator & **getInstance** ()

Private Attributes

- std::mt19937 **generator_**
- std::uniform_real_distribution< float > **dis_0_1**

5.17.1 Detailed Description

Random number generator using Mersenne Twister and std::uniform_real_distribution

5.17.2 Member Function Documentation

5.17.2.1 random()

```
template<int a, int b>
float genetic_tetris::RandomNumberGenerator::random ( )  [inline]
```

Return random value from given range

Template Parameters

a	min value
b	max value

Returns

random number

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/AI/random_number_generator.hpp

5.18 genetic_tetris::SoundManager Class Reference

Classes

- struct SoundProperties

Public Types

- enum **Sound** { **TETRIS_THEME**, **CLICK**, **HARD_DROP**, **ROW_CLEARED** }

Public Member Functions

- **SoundManager** (const SoundManager &)=delete
- SoundManager **operator=** (const SoundManager &)=delete
- void **play** (Sound sound)

Static Public Member Functions

- static SoundManager & **getInstance** ()

Private Types

- typedef struct genetic_tetris::SoundManager::SoundProperties **SoundProperties**

Private Member Functions

- void **garbageCollector** ()

Static Private Member Functions

- static std::unordered_map< Sound, SoundProperties > & **getSounds** ()

Private Attributes

- std::vector< std::unique_ptr< sf::Sound > > **playing_sounds_**

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/sound_manager.hpp

5.19 genetic_tetris::SoundManager::SoundProperties Struct Reference

Public Member Functions

- **SoundProperties** (std::string path, float volume, bool loop=false)
- **SoundProperties** (const SoundProperties &sound_properties)

Public Attributes

- std::string **path**
- float **volume**
- bool **loop**
- std::unique_ptr< sf::SoundBuffer > **buffer**

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/sound_manager.hpp

5.20 genetic_tetris::Subject Class Reference

`#include <utils.hpp>`

Inheritance diagram for genetic_tetris::Subject:

Public Member Functions

- void **addObserver** (Observer *o)
- void **notifyObservers** (EventType e)

Private Attributes

- std::vector< Observer * > **obs_**

5.20.1 Detailed Description

Subject class for Observer.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/utils.hpp

5.21 genetic_tetris::Tetris Class Reference

Inheritance diagram for genetic_tetris::Tetris:

Public Types

- using **Position** = std::pair< int, int >
- using **Grid** = std::vector< std::vector< Tetromino::Color > >

Public Member Functions

- **Tetris** (bool disable_drop_scores=false)
- bool **tick** (bool is_soft_drop=false)
- void **shiftLeft** ()
- void **shiftRight** ()
- void **hardDrop** (bool tick_after_drop=true)
- void **rotateCW** ()
- void **rotateCCW** ()
- Grid **getRawGrid** () const
- Grid **getDisplayGrid** () const
- std::string **toString** () const
- bool **isFinished** () const
- unsigned int **getScore** () const
- unsigned int **getLevel** () const
- unsigned int **getLevelProgress** () const
- double **getLevelSpeed** () const
- unsigned int **getLastTickClearedRowsCount** () const
- std::deque< Tetromino > **getTetrominoQueue** () const

Static Public Attributes

- static const int **GRID_WIDTH** = 10
- static const int **GRID_VISIBLE_HEIGHT** = 20
- static const int **GRID_FULL_HEIGHT** = 40
- static constexpr Position **TETROMINO_INITIAL_POS**
- static const int **MAX_LEVEL** = 15
- static const int **LINES_PER_LEVEL** = 10
- static const int **SCORE_SINGLE** = 100
- static const int **SCORE_DOUBLE** = 300
- static const int **SCORE_TRIPLE** = 500
- static const int **SCORE_TETRIS** = 800
- static const int **SCORE_SOFT_DROP** = 1
- static const int **SCORE_HARD_DROP** = 2

Protected Member Functions

- virtual void **generateTetromino** ()

Private Member Functions

- bool **isValidPosition** (Position tetromino_position) const
- Position **getHardDropPosition** () const
- void **clearLines** ()
- void **addClearedLinesScore** ()
- void **addProgress** ()
- void **calculateLevelSpeed** ()
- void **rotate** (bool ccw)

Private Attributes

- TetrominoGenerator **generator_**
- Tetromino **tetromino_**
- Position **tetromino_position_**
- Grid **grid_**
- bool **is_finished_**
- unsigned int **score_**
- unsigned int **level_**
- unsigned int **level_progress_**
- double **level_speed_**
- unsigned int **cleared_rows_**
- bool **drop_scores_disabled_**

5.21.1 Member Data Documentation

5.21.1.1 TETROMINO_INITIAL_POS

```
constexpr Position genetic_tetris::Tetris::TETROMINO_INITIAL_POS  [static], [constexpr]
```

Initial value:

```
= {(GRID_WIDTH / 2) - 2,
                                                    (GRID_FULL_HEIGHT / 2) - 1}
```

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/tetris/tetris.hpp

5.22 genetic_tetris::TetrisBoard Class Reference

```
#include <gui_utils.hpp>
```

Classes

- struct TileProperties

Public Member Functions

- **TetrisBoard** (const sf::Vector2f &position, const sf::Vector2i &board_tile_count, const TileProperties &tile↩ _prop)
- void **setState** (const Tetris::Grid &tetris_grid)
- void **setTetrominoQueue** (const std::deque< Tetromino > &queue)
- void **draw** (sf::RenderWindow &window)
- void **reset** ()
- bool **isStateFinished** () const
- void **setStateFinished** (bool finished)

Private Types

- using **Board** = std::vector< std::vector< sf::RectangleShape > >

Private Attributes

- const sf::Color **FINISHED_HUE_CHANGE_** = sf::Color(50, 50, 50, 0)
- bool **state_finished_**
- Board **board_**
- sf::Vector2i **board_tile_count_**

5.22.1 Detailed Description

Class used to display tetris grid and next tetromino panel.

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/gui/gui_utils.hpp

5.23 genetic_tetris::Tetromino Class Reference

Public Types

- enum **Color** {
  EMPTY, **CYAN**, **YELLOW**, **PURPLE**,
  GREEN, **RED**, **BLUE**, **ORANGE**,
  GHOST }
- enum **Shape** {
  NO_SHAPE, **I**, **O**, **T**,
  S, **Z**, **J**, **L** }
- using **Pivot** = std::pair< double, double >
- using **Square** = std::pair< int, int >
- using **Squares** = std::vector< Square >
- using **Rotations** = std::vector< Squares >

Public Member Functions

- **Tetromino** (Color color, Shape shape, Pivot pivot, const Squares &squares)
- **Tetromino** (const Tetromino &tetromino)=default
- Tetromino & **operator=** (const Tetromino &tetromino)=default
- void **rotateCW** ()
- void **rotateCCW** ()
- Color **getColor** () const
- Shape **getShape** () const
- const Squares & **getSquares** () const
- int **getCurrentRotation** () const
- std::string **toString** () const

Static Public Member Functions

- static Squares **rotate** (const Squares &squares, const Pivot &pivot, double rad)

Private Attributes

- Color **color_**
- Shape **shape_**
- Rotations **rotations_**
- int **current_rotation_**

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/tetris/tetromino.hpp

5.24 genetic_tetris::TetrominoGenerator Class Reference

Public Member Functions

- Tetromino **getNextTetromino** ()
- std::deque< Tetromino > **getQueue** () const

Static Public Member Functions

- static const std::vector< Tetromino > & **getTetrominoes** ()

Static Public Attributes

- static const unsigned int **QUEUE_LENGTH** = 4

Private Member Functions

- void **generateTetrominoes** ()

Private Attributes

- std::deque< Tetromino > **queue_**

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/tetris/tetromino_generator.hpp

5.25 genetic_tetris::TetrisBoard::TileProperties Struct Reference

Public Member Functions

- **TileProperties** (float size, float padding)

Public Attributes

- float **size**
- float **padding**
- float **padded_size**

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

- /home/damian/Desktop/Laby/ZPR-lab/genetic-tetris/project/include/gui/gui_utils.hpp