Tetris

Generated by Doxygen 1.10.0

Chapter 1

User Manual

Main Menu

Upon starting the application, the user is greeted with a menu screen that has three buttons: Start Tetris, Leaderboards, and Exit. The Start button will take the user to the game screen, where they can play Tetris. The Leaderboards button will take the user to the leaderboard screen, where they can view the high scores set in the game, stored locally. The Exit button will close the application and save the scores from the leaderboards to a local file, so they don't get lost.

Game Screen

The game screen is where the user can play Tetris. On the left hand side of the screen is the game board, where the Tetris pieces will fall. The user can move the pieces left or right using the arrow keys, rotate them left or right using the A or D keys, or drop them to the bottom of the board using the space key.

On the right hand side of the screen at the top the player can see their current statistics. These are: their score, the number of lines they have cleared, the level they are currently on, and how many lines they need to clear to advance to the next level. Below this is the next piece that will fall, so the player can plan ahead.

At the bottom of the screen is a back button, which will take the user back to the main menu and discard the current game.

Rules and Scoring

The goal of Tetris is to clear lines by filling them with blocks. When a line is filled, it will disappear and the blocks above it will fall down. The player will receive points for each line they clear, with more points being awarded for clearing multiple lines at once. The player will also receive bonus points for using the drop key to drop a piece faster. The game will end if a piece overhangs the top of the board, at which point the player will be asked to enter their name to be put on the leaderboards. If the name field is left empty, the score is discarded.

As the player clears more lines, they will advance to higher levels. The pieces will fall faster as the player climbs in levels, making it more difficult to clear lines. The player will also receive more points for clearing lines at higher levels. There are a total of 6 levels, the player starts on level 1, and after 10 times the level count amount of lines have been cleared, the player advances to the next level.

Scoring is as follows, where s is the current game level, and d is the height the player dropped the piece from (up to 5, or 0 if the drop key was not used):

- 1 line: $40 * (s^2 + d)$
- 2 lines: $100 * (s^2 + d)$
- 3 lines: $300 * (s^2 + d)$
- 4 lines: $1200 * (s^2 + d)$

Leaderboard Screen

The leaderboard screen displays the best scores that have been set in the game on the current device. The scores are displayed in descending order, with the highest score at the top. The player can see the name of the player who set the score and the score itself.

If there are less scores than the maximum amount of scores that can be displayed, the remaining rows will have the name *<empty>* and the score *0*.

At the bottom of the screen is a back button, which will take the user back to the main menu.

Chapter 2

Class Diagram

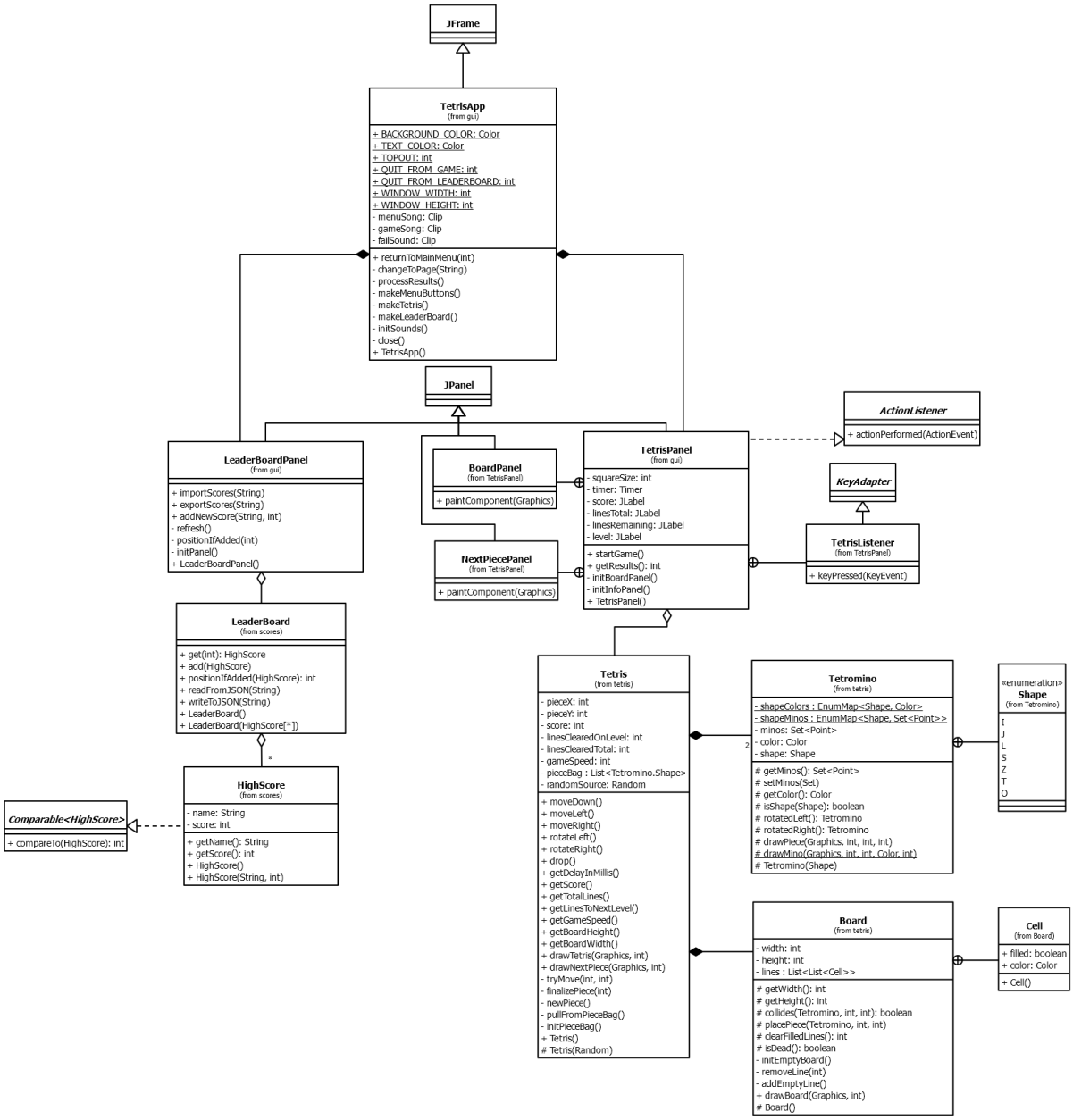


Figure 2.1 UML Class Diagram

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

Class Documentation

5.1 tetris.Board Class Reference

Public Member Functions

- void drawBoard (Graphics g, int squareSize)

Protected Member Functions

- Board ()
- int getWidth ()
- int getHeight ()
- boolean collides (Tetromino piece, int x, int y)
- void placePiece (Tetromino piece, int x, int y)
- int clearFilledLines ()
- boolean isDead ()

5.1.1 Detailed Description

The Board class represents the game board in Tetris, which is a grid of cells that can be filled by Tetrominos.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Board()

```
tetris.Board.Board ( )  [protected]
```

Constructs a new Board object with the default width and height, 10x24 as in the original Tetris game. Adds 4 hidden rows at the top of the board to allow the spawning of new Tetrominos and for death to occur.

5.1.3 Member Function Documentation

5.1.3.1 clearFilledLines()

```
int tetris.Board.clearFilledLines ( )  [protected]
```

Clears all lines that are completely filled with Tetrominos, removing them from the board which results in the pieces above them being dropped down, and adding new empty lines at the top to keep the height of the board constant.

Returns

the number of lines that were cleared

5.1.3.2 collides()

```
boolean tetris.Board.collides (
            Tetromino piece,
            int x,
            int y )  [protected]
```

Checks if the given Tetromino piece collides with the current state of the board at the given (x, y) origin position, using the relative coordinates of its minos.

Parameters

piece	the Tetromino piece to check for collision
x	the x-coordinate of the origin for the Tetromino's position
y	the y-coordinate of the origin for the Tetromino's position

Returns

true if the piece collides with the board or is out of bounds, false otherwise

5.1.3.3 drawBoard()

```
void tetris.Board.drawBoard (
            Graphics g,
            int squareSize )
```

Draws the board on the given Graphics object, using the given square size to scale the cells. Origin is at the top-left corner, with the x-axis increasing to the right and the y-axis increasing downwards.

Parameters

g	
squareSize	

5.1.3.4 getHeight()

```
int tetris.Board.getHeight ( )  [protected]
```

Returns the height of the board in cells, excluding the 4 hidden lines at the top.

Returns

the playable height of the board

5.1.3.5 getWidth()

```
int tetris.Board.getWidth ( )  [protected]
```

Returns the width of the board in cells.

Returns

the width of the board

5.1.3.6 isDead()

```
boolean tetris.Board.isDead ( )  [protected]
```

Checks if any lines above the playable height of the board have any filled cells in them, which means the stack is too high and the game is over.

Returns

true if the game is over, false otherwise

5.1.3.7 placePiece()

```
void tetris.Board.placePiece (
            Tetromino piece,
            int x,
            int y )  [protected]
```

Places the given Tetromino piece on the board at the given (x, y) origin position, using the relative coordinates of its minos to fill the cells with the Tetromino's color. This method does not check for collisions, it should only be called after checking with collides().

Parameters

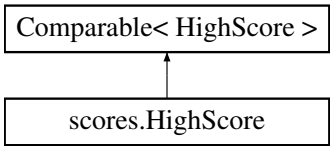
piece	the Tetromino piece to place on the board
x	the x-coordinate of the origin for the Tetromino's position
y	the y-coordinate of the origin for the Tetromino's position

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

- Tetris/src/tetris/Board.java

5.2 scores.HighScore Class Reference

Inheritance diagram for scores.HighScore:



Public Member Functions

- HighScore ()
- HighScore (String n, int s)
- String getName ()
- int getScore ()
- int compareTo (HighScore h)

5.2.1 Detailed Description

The HighScore class represents a high score with a name and a score. It implements the Comparable<HighScore> interface to allow natural ordering by score.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 HighScore() [1/2]

```
scores.HighScore.HighScore ( )
```

Constructs a new HighScore object with an empty name and a score of 0.

5.2.2.2 HighScore() [2/2]

```
scores.HighScore.HighScore (
            String n,
            int s )
```

Constructs a new HighScore object with the specified name and score.

Parameters

n	the name of the player
s	the score achieved by the player

5.2.3 Member Function Documentation

5.2.3.1 compareTo()

```
int scores.HighScore.compareTo (
            HighScore h )
```

Compares this HighScore object with another HighScore object by score.

Parameters

h	the HighScore object to compare to

Returns

a negative integer, zero, or a positive integer as this HighScore's score is less than, equal to, or greater than the specified HighScore's score

5.2.3.2 getName()

```
String scores.HighScore.getName ( )
```

Returns the name of the player who achieved the high score. If the name is empty or the score is 0, returns "<empty>", as these scores don't need to be recorded.

Returns

the name of the player

5.2.3.3 getScore()

```
int scores.HighScore.getScore ( )
```

Returns the score achieved by the player. If the name is empty, returns 0, as these scores don't need to be recorded.

Returns

the score achieved by the player

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

- Tetris/src/scores/HighScore.java

5.3 scores.LeaderBoard Class Reference

Public Member Functions

- LeaderBoard ()
- LeaderBoard (HighScore... scores)
- HighScore get (int index)
- void add (HighScore s)
- int positionIfAdded (int score)
- void readFromJSON (String filename)
- void writeToJSON (String filename)

5.3.1 Detailed Description

The LeaderBoard class represents a leaderboard with a list of high scores.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 LeaderBoard() [1/2]

```
scores.LeaderBoard.LeaderBoard ( )
```

Constructs a new LeaderBoard object with an empty list of high scores.

5.3.2.2 LeaderBoard() [2/2]

```
scores.LeaderBoard.LeaderBoard (
            HighScore...  scores )
```

Constructs a new LeaderBoard object with the specified high scores, sorting them in descending order.

Parameters

scores	the high scores to add to the leaderboard

5.3.3 Member Function Documentation

5.3.3.1 add()

```
void scores.LeaderBoard.add (
            HighScore s )
```

Adds a new high score to the leaderboard, sorting the list in descending order. If the score is invalid, it is not added.

Parameters

s	the high score to add

5.3.3.2 get()

```
HighScore scores.LeaderBoard.get (
            int index )
```

Returns the high score at the specified index in the leaderboard. If the given index is higher than the number of high scores, returns an empty high score.

Parameters

index	the index of the high score to return

Returns

the high score at the specified index

5.3.3.3 positionIfAdded()

```
int scores.LeaderBoard.positionIfAdded (
            int score )
```

Returns the position the given score would have in the leaderboard, if added to it.

Parameters

score	the score to check

Returns

the position the score would have in the leaderboard

5.3.3.4 readFromJSON()

```
void scores.LeaderBoard.readFromJSON (
            String filename )
```

Reads high scores from a JSON file and adds them to the leaderboard. The file should only contain valid high scores, in descending order.

Parameters

filename	the name of the file to read from

5.3.3.5 writeToJSON()

```
void scores.LeaderBoard.writeToJSON (
            String filename )
```

Writes the high scores to a JSON file. The file will contain the high scores in descending order.

Parameters

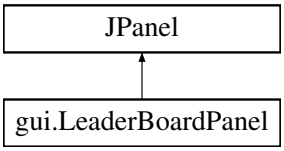
filename	the name of the file to write to

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

- Tetris/src/scores/LeaderBoard.java

5.4 gui.LeaderBoardPanel Class Reference

Inheritance diagram for gui.LeaderBoardPanel:



Public Member Functions

- LeaderBoardPanel (TetrisApp p)
- void importScores (String filename)
- void exportScores (String filename)
- void addNewScore (String name, int score)
- int positionIfAdded (int score)

5.4.1 Detailed Description

The LeaderBoardPanel class is responsible for displaying the leaderboard to the user.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 LeaderBoardPanel()

```
gui.LeaderBoardPanel.LeaderBoardPanel (
            TetrisApp p )
```

Constructs a new LeaderBoardPanel with the given parent TetrisApp.

Parameters

p	the parent TetrisApp object

5.4.3 Member Function Documentation

5.4.3.1 addNewScore()

```
void gui.LeaderBoardPanel.addNewScore (
            String name,
            int score )
```

Adds a new entry to the leaderboard, and refreshes the panel to display it.

Parameters

name	the name of the player
score	the score of the player

5.4.3.2 exportScores()

```
void gui.LeaderBoardPanel.exportScores (
            String filename )
```

Exports high scores to a JSON file.

Parameters

filename	the name of the file to export to

5.4.3.3 importScores()

```
void gui.LeaderBoardPanel.importScores (
            String filename )
```

Imports high scores from a JSON file, and refreshes the panel to display them.

Parameters

filename	the name of the file to import from

5.4.3.4 positionIfAdded()

```
int gui.LeaderBoardPanel.positionIfAdded (
            int score )
```

Returns the position that the given score would be at if added to the leaderboard. Interface for the LeaderBoard object.

Parameters

score	the score to check

Returns

the position the score would be at if added

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

- Tetris/src/gui/LeaderBoardPanel.java

5.5 tetris.Tetromino.Shape Enum Reference

Public Attributes

- **I**
- **J**
- **L**
- **S**
- **Z**
- **T**
- **O**

5.5.1 Detailed Description

The Shape enum represents the different types of tetromino shapes used in Tetris. Each shape is represented by its 1-letter name: I, J, L, S, Z, T, O.

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

- Tetris/src/tetris/Tetromino.java

5.6 tetris.Tetris Class Reference

Public Member Functions

- Tetris ()
- boolean moveDown ()
- void moveLeft ()
- void moveRight ()
- void rotateLeft ()
- void rotateRight ()
- void drop ()
- int getDelayInMillis ()
- int getScore ()
- int getTotalLines ()
- int getLinesToNextLevel ()
- int getGameSpeed ()
- int getBoardHeight ()
- int getBoardWidth ()
- void drawTetris (Graphics g, int squareSize)
- void drawNextPiece (Graphics g, int squareSize)

Protected Member Functions

- Tetris (Random r)

5.6.1 Detailed Description

The Tetris class represents the game logic for a Tetris game, including the board, falling piece, and scoring.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 Tetris() [1/2]

```
tetris.Tetris.Tetris ( )
```

Constructs a new Tetris object with a new empty Board. Sets all performance metrics to 0 and the game speed to 1. Initializes the current and next pieces with the first two pieces from the pieceBag. Sets the random source to a built in Random object.

5.6.2.2 Tetris() [2/2]

```
tetris.Tetris.Tetris (
            Random r )  [protected]
```

Constructs a new Tetris object with a new empty Board. Sets all performance metrics to 0 and the game speed to 1. Initializes the current and next pieces with the first two pieces from the pieceBag. Sets the random source to a given Random object. Protected scope because this is used for testing.

5.6.3 Member Function Documentation

5.6.3.1 drawNextPiece()

```
void tetris.Tetris.drawNextPiece (
            Graphics g,
            int squareSize )
```

Calls the draw function of the next piece, to draw it in a preview window. The preview window should be a 5x5 square, the drawing is done in a way where the piece is centered in the window.

Parameters

g	the Graphics object to draw on
squareSize	the size of each square in pixels

5.6.3.2 drawTetris()

```
void tetris.Tetris.drawTetris (
```

```
Graphics g,
int squareSize )
```

Calls the draw functions of both the board and the current piece to draw the game state to the given Graphics object.

Parameters

g	the Graphics object to draw on
squareSize	the size of each square in pixels

5.6.3.3 drop()

```
void tetris.Tetris.drop ( )
```

Performs a "hard-drop", which moves the current piece down as far as it can go without colliding with the board. This is guaranteed to result in a piece landing, so it will finalize the piece and generate a new one. Extra points are awarded for the height of the drop, up to 5 cells.

5.6.3.4 getBoardHeight()

```
int tetris.Tetris.getBoardHeight ( )
```

Returns the height of the board in cells.

Returns

the height of the board

5.6.3.5 getBoardWidth()

```
int tetris.Tetris.getBoardWidth ( )
```

Returns the width of the board in cells.

Returns

the width of the board

5.6.3.6 getDelayInMillis()

```
int tetris.Tetris.getDelayInMillis ( )
```

Returns the delay in milliseconds between the steps of automatic falling, based on the current game speed. The delay is given by the time it takes for a piece to drop from its spawning point to the bottom of the board. This time for each level is: 10s, 6s, 3s, 1.5s, 1s, 0.4s.

Returns

the delay in milliseconds between steps of falling

5.6.3.7 getGameSpeed()

```
int tetris.Tetris.getGameSpeed ( )
```

Returns the current level.

Returns

the current level

5.6.3.8 getLinesToNextLevel()

```
int tetris.Tetris.getLinesToNextLevel ( )
```

Returns the number of lines remaining to clear before the next level is reached. The number of lines to clear is 10 times the current game speed minus the number of lines cleared in the current level. The player can not progress further from level 6, so the number of lines to clear is always 1.

Returns

the number of lines remaining to clear before the next level

5.6.3.9 getScore()

```
int tetris.Tetris.getScore ( )
```

Returns the current score of the game.

Returns

the current score

5.6.3.10 getTotalLines()

```
int tetris.Tetris.getTotalLines ( )
```

Returns the number of lines cleared in the current level.

Returns

the number of lines cleared in the current level

5.6.3.11 moveDown()

```
boolean tetris.Tetris.moveDown ( )
```

Attempts to move the current piece down by one cell. If the piece cannot move down, it will put the piece on the board, add points, and generate a new piece.

Returns

true if the piece was successfully moved or if it landed and the game continues, false if the game is over

5.6.3.12 moveLeft()

```
void tetris.Tetris.moveLeft ( )
```

Attempts to move the current piece left by one cell. This can not result in a piece landing, so no finalization is needed.

5.6.3.13 moveRight()

```
void tetris.Tetris.moveRight ( )
```

Attempts to move the current piece right by one cell. This can not result in a piece landing, so no finalization is needed.

5.6.3.14 rotateLeft()

```
void tetris.Tetris.rotateLeft ( )
```

Attempts to rotate the current piece left (counter-clockwise) by 90 degrees. This can not result in a piece landing, so no finalization is needed.

5.6.3.15 rotateRight()

```
void tetris.Tetris.rotateRight ( )
```

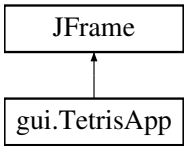
Attempts to rotate the current piece right (clockwise) by 90 degrees. This can not result in a piece landing, so no finalization is needed.

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

- Tetris/src/tetris/Tetris.java

5.7 gui.TetrisApp Class Reference

Inheritance diagram for gui.TetrisApp:



Public Member Functions

- TetrisApp ()
- void returnToMainMenu (int reason)

Static Public Member Functions

- static void main (String[ ] args)

Static Public Attributes

- static final Color BACKGROUND_COLOR = new Color(40, 42, 54).darker()
- static final Color TEXT_COLOR = new Color(248, 248, 242)
- static final int TOPOUT = 0
- static final int QUIT_FROM_GAME = 1
- static final int QUIT_FROM_LEADERBOARD = 2

5.7.1 Detailed Description

The TetrisApp class is the main class of the game.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 TetrisApp()

```
gui.TetrisApp.TetrisApp ( )
```

Constructs a new TetrisApp object with the default window settings. Initializes the sounds used in the app. Creates the 3 main screens: the menu, the game, and the leaderboard, which are managed by a CardLayout, to allow easy switching based on their names. The default closing operation does nothing, so the app can handle it manually, which is done by using a WindowAdapter to call the close() method.

5.7.3 Member Function Documentation

5.7.3.1 main()

```
static void gui.TetrisApp.main (
            String[] args )  [static]
```

main method that initializes some UI settings and creates a new TetrisApp object. Look and feel is set to FlatLaf, from https://www.formdev.com/flatlaf/

5.7.3.2 returnToMainMenu()

```
void gui.TetrisApp.returnToMainMenu (
            int reason )
```

Returns to the main menu screen with the given reason. TOPOUT: the player topped out (died) in the game. QUIT_FROM_GAME: the player quit from the game without dying. QUIT_FROM_LEADERBOARD: the player is returning from the leaderboard.

Parameters

reason	the reason for returning to the main menu

5.7.4 Member Data Documentation

5.7.4.1 BACKGROUND_COLOR

```
final Color gui.TetrisApp.BACKGROUND_COLOR = new Color(40, 42, 54).darker()  [static]
```

The background color of the app.

5.7.4.2 QUIT_FROM_GAME

```
final int gui.TetrisApp.QUIT_FROM_GAME = 1  [static]
```

The constant representing the player quitting from the game without dying.

5.7.4.3 QUIT_FROM_LEADERBOARD

```
final int gui.TetrisApp.QUIT_FROM_LEADERBOARD = 2  [static]
```

The constant representing the player returning from the leaderboard.

5.7.4.4 TEXT_COLOR

```
final Color gui.TetrisApp.TEXT_COLOR = new Color(248, 248, 242)  [static]
```

The text color of the app.

5.7.4.5 TOPOUT

```
final int gui.TetrisApp.TOPOUT = 0  [static]
```

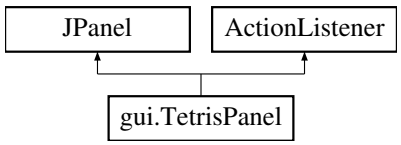
The constant representing the player topping out (dying) in the game.

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

- Tetris/src/gui/TetrisApp.java

5.8 gui.TetrisPanel Class Reference

Inheritance diagram for gui.TetrisPanel:



Public Member Functions

- TetrisPanel (TetrisApp p)
- void startGame ()
- void actionPerformed (ActionEvent e)
- int getResults ()

5.8.1 Detailed Description

The TetrisPanel class is responsible for interfacing between the game and the user. It runs the game itself through a Tetris object and collects inputs from the player.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 TetrisPanel()

```
gui.TetrisPanel.TetrisPanel (
            TetrisApp p )
```

Constructs a new TetrisPanel with the given parent TetrisApp.

Parameters

p	the parent TetrisApp object

5.8.3 Member Function Documentation

5.8.3.1 actionPerformed()

```
void gui.TetrisPanel.actionPerformed (
            ActionEvent e )
```

The main game loop that runs the game logic and updates the display. Tries moving the current piece down, if moveDown() returns false, we know the game is over, so it stops the timer and returns to the main menu with the TOPOUT flag. Updates the timer delay and all the info labels with the current game state each tick. Calls the repaint() method to make sure the board and next piece is updated.

5.8.3.2 getResults()

```
int gui.TetrisPanel.getResults ( )
```

Returns the end results of the game.

Returns

the score of the game

5.8.3.3 startGame()

```
void gui.TetrisPanel.startGame ( )
```

Calculates squareSize and initializes the game board and info panels using it. Starts the game timer and sets the focus to the panel to be able to receive key inputs. The ActionListener for the timer is this panel itself.

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

- Tetris/src/gui/TetrisPanel.java

5.9 tetris.Tetromino Class Reference

Classes

- enum Shape

Protected Member Functions

- Tetromino (Shape shape)
- Set< Point > getMinos ()
- void setMinos (Set< Point > newMinos)
- Color getColor ()
- boolean isShape (Shape s)
- Tetromino rotatedLeft ()
- Tetromino rotatedRight ()
- void drawPiece (Graphics g, int x, int y, int squareSize)

Static Protected Member Functions

- static void drawMino (Graphics g, int x, int y, Color c, int size)

5.9.1 Detailed Description

The Tetromino class represents a Tetris piece with a specific shape, color, and a set of (four) minos.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 Tetromino()

```
tetris.Tetromino.Tetromino (
            Shape shape )  [protected]
```

Constructs a new Tetromino object with the specified shape, setting the minos and color accordingly.

Parameters

shape	the shape of the Tetromino

5.9.3 Member Function Documentation

5.9.3.1 drawMino()

```
static void tetris.Tetromino.drawMino (
            Graphics g,
            int x,
            int y,
            Color c,
            int size )  [static], [protected]
```

Draws a single mino on the graphics context at the specified location and size. Draws a darker border around the mino.

Parameters

g	the graphics context to draw on
x	the x-coordinate of the top-left corner of the mino
y	the y-coordinate of the top-left corner of the mino
c	the color of the mino
size	the size of the mino in pixels

5.9.3.2 drawPiece()

```
void tetris.Tetromino.drawPiece (
            Graphics g,
            int x,
            int y,
            int squareSize )  [protected]
```

Draws the Tetromino on the given graphics context at the specified location and size. Assumes the minos are relative to the top-left corner of the piece, increasing height downwards.

Parameters

g	the graphics context to draw on
x	the x-coordinate of the top-left corner of the piece
y	the y-coordinate of the top-left corner of the piece
squareSize	the size of each square in pixels

5.9.3.3 getColor()

```
Color tetris.Tetromino.getColor ( )  [protected]
```

Returns the color of the Tetromino.

Returns

the color of the Tetromino

5.9.3.4 getMinos()

```
Set< Point > tetris.Tetromino.getMinos ( )  [protected]
```

Returns the minos of the Tetromino.

Returns

a Set of points representing the minos

5.9.3.5 isShape()

```
boolean tetris.Tetromino.isShape (
            Shape s )  [protected]
```

Checks if the Tetromino has the given shape.

Parameters

s	the shape to check against

Returns

true if the Tetromino has the given shape, false otherwise

5.9.3.6 rotatedLeft()

```
Tetromino tetris.Tetromino.rotatedLeft ( )  [protected]
```

Rotates the Tetromino 90 degrees to the left (counter-clockwise).

Returns

a new Tetromino object representing the rotated piece

5.9.3.7 rotatedRight()

```
Tetromino tetris.Tetromino.rotatedRight ( )  [protected]
```

Rotates the Tetromino 90 degrees to the right (clockwise).

Returns

a new Tetromino object representing the rotated piece

5.9.3.8 setMinos()

```
void tetris.Tetromino.setMinos (
            Set< Point > newMinos )  [protected]
```

Sets the minos of the Tetromino to the given set of points.

Parameters

newMinos	the new set of minos

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

- Tetris/src/tetris/Tetromino.java

Index