Tetris on Processing – JuanGg on December 2018

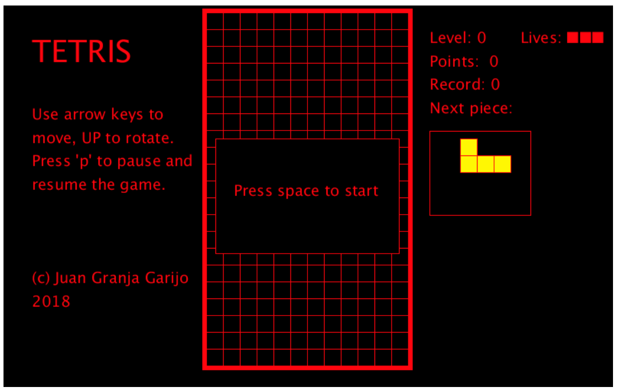
1. Introduction:

The game at hand is a fully featured Tetris game written in the Processing environment. It is programmed to behave just like any regular Tetris would.

2. Operation:

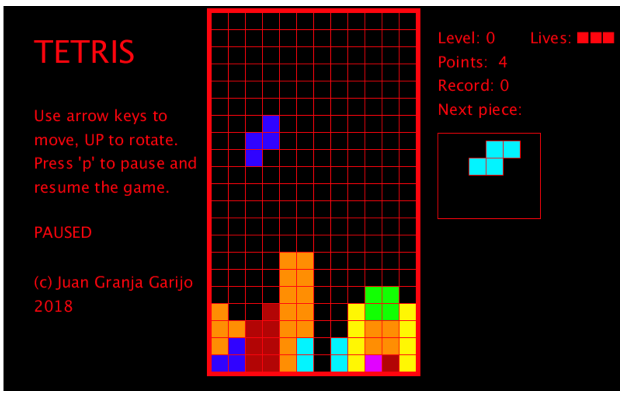
When running the game, the user is prompted with the screen shown in **Figure 1** and asked to press the "space" key to start playing. Left hand side of the screen gives the user information on how to play the game, while right hand side contains counters for all variables relevant to the it. This include the current level, current points, last record and lives left. All these start as 0, but the lives counter, which starts at 3. In addition, a preview of the next piece which is to appear is also shown. In the middle of the screen, the main grid in which the game takes place is shown.

After pressing space, the game moves to the game screen shown in **Figure 2**. Random pieces appear on top and the user can move and rotate them using the arrow keys, just like in a regular Tetris. Each completed line is erased and the remaining ones moved down, summing points depending on the current level. If the pieces reach the top, the grid is blanked and one life is subtracted. The user is able to pause and resume the game pressing the 'p' key.



TETRIS
Use arrow keys to move, UP to rotate. Press 'p' to pause and resume the game.
(c) Juan Granja Garijo 2018
Press space to start
Level: 0
Lives:
Points: 0
Record: 0
Next piece:

Figure 1: Welcome screen



TETRIS
Use arrow keys to move, UP to rotate. Press 'p' to pause and resume the game.
PAUSED
(c) Juan Granja Garijo 2018
Level: 0
Lives:
Points: 4
Record: 0
Next piece:

Figure 2: Game screen

If there are no lives left, game ends and record is saved, shown on **Figure 3**. Each 10 points, the game goes up one level, the color theme changes accordingly and so does the speed that the pieces fall at. The lives counter increases in one to a maximum of 3 as well.

There are a maximum of 4 levels, after which the game ends with the screen in **Figure 4**.

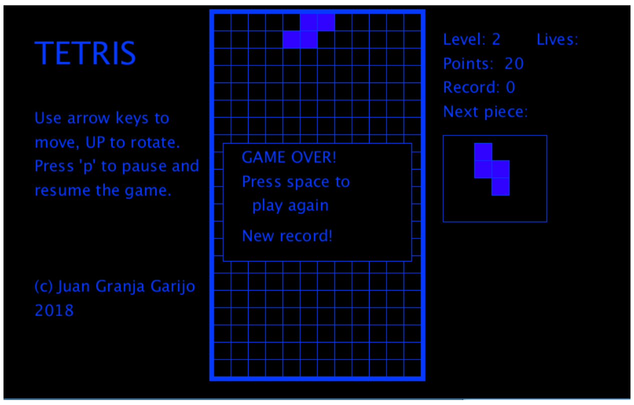

TETRIS
Use arrow keys to move, UP to rotate. Press 'p' to pause and resume the game.
(c) Juan Granja Garijo 2018
GAME OVER! Press space to play again
New record!
Level: 2
Lives:
Points: 20
Record: 0
Next piece:

Figure 3: Game over screen

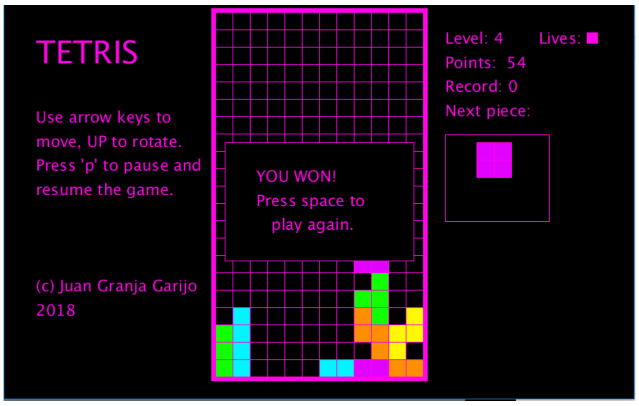

TETRIS
Use arrow keys to move, UP to rotate. Press 'p' to pause and resume the game.
(c) Juan Granja Garijo 2018
YOU WON! Press space to play again.
Level: 4
Lives:
Points: 54
Record: 0
Next piece:

Figure 4: Game end screen

3. Game logic brief summary:

The core of the game functionality is based around the Piece class. This class has variables such as the kind of piece, its orientation and its XY position on the main game grid. It also contains an int array of all the available piece colors, and a 4-dimension array in which all piece information is stored in 4x4 Boolean matrices. Every kind of piece has four associated 4x4 matrices, one per orientation. During the game, these matrices are what gets moved on the main grid. This main grid has a two-dimensional array associated which contains the colors of all squares of the grid.

To begin with, a `nextPiece` object is created, with random attributes. This object is shown on the right-hand side of the screen as a preview of the next piece to appear. Then this object is copied to a `currentPiece` object and a new `nextPiece` is created.

The `currentPiece` object is the actual piece that falls from top to bottom. The user can change its position and orientation, and the program continuously checks whether if the move that is about to be made is allowable or not. This is done by a method that checks if the piece is going outside the grid or collides with squares filled with a color different than the background (black). When the piece cannot go down further, its squares get copied to the main grid and the `currentPiece` object is discarded. A new piece appears on the top and the process repeats.

While this happens, another method checks for complete rows, and if found, it moves all upper rows down by copying square colors one by one, one position down. It then adds points to the points counter. Please refer to the code for more details.

4. Legal:

This is an original work of JuanGg based on the Tetris game. This was done from scratch, not taking any reference code or algorithm as a starting point. No copyright infringement is intended.