Finishing Tetrominos

A few hints on how to finish the Tetrominos game.

At this point, you should have a well-organized working program that lets you draw colored blocks on the screen. As this is a challenge project, the rest is up to you. I recommend breaking the project down into manageable chunks, setting some intermediate goals, and reaching those goals one at a time. There is no hurry – take plenty of time to think through each step, and test it before going on to the next item.

Here are some things you'll need to do:

1. Create the falling pieces, which are made up of multiple blocks. I wrote a class `Piece` to keep track of information for a small group of blocks.

2. Each block needs a physical location within the piece – the layout of the piece. I created a class `Location` that simply stores an x, y pair. A `Piece` then contains a list of locations.

3. Each Piece needs to be able to fall, so each piece needs to know where it is on the screen. I added an instance variable `position` of type `Location` to the Piece class. I wrote a method `translate` in the Piece class. To actually make the piece fall, I had a line of code `activePiece.translate(0, -1);` in the `nextTurn` method of the Board class. Since the `nextTurn` method is called every second or so by the Tetrominos user interface, this causes the active piece to fall downwards one step at a time.

4. The blocks should translate left and write as the user presses the `s` and `f` keys on the keyboard. When those keys are pressed, the `slide` method of Board.java is called. Finish the `slide` method.

5. Add rotation in response to key presses. This step is trickiest, and you might leave it until last, since you can write a working game without it. You'll need to figure out how locations within a piece change on rotation.

6. Check for collisions between the falling *active* piece and the bottom of the screen, and blocks that are already sitting on the bottom of the screen. When there is a collision, copy block locations from the Piece to the Board to represent the leftover blocks, and then create a new active piece.
7. Check for complete rows of blocks in in the board, and delete them.

Good luck, and happy coding!