Ass7Game - Space Invaders

Code Description

The changed interfaces and classes in my code:

- Ass7game - holds the main menu of the game. I took out the sub-menu from the previous assignment and changed the GUI's title to "Space Invaders".
- Block - Block is a collidable rectangle. I added a Boolean private to this class, called "isInvader", and a getter and a setter for it, so when the block is hit and GameLevel tries to remove it from its sprites, it will know exactly how. I also added a method called "moveRect" that will reset the block's location (in order to move each invader).
- GameFlow - I changed the game loop so that each level in the game is the same except for different initial speed of the invaders, and that the game will only end by the player loosing.
- GameLevel - I've added the aforementioned privates:
  - private Invaders invaders - the level's invaders.
  - private ArrayList<Ball> bullets - all the bullets that were shot during this level will have to be removed if it is restarting or cleared.
  - private int levelNumber and private int invadersSpeed - these are different between the levels.
  - private boolean allInvadersDied - indicates that the level is cleared.
  - private SpaceShip spaceship - the player's controller to the game.

  I have added or changed the aforementioned methods:
  - As mentioned, I changed the "removeSprite" method to check if a sprite is an invader and to remove it accordingly.
  - setShield() - the shield is simply a group of tiny blocks. Each block that is being hit by a bullet (Ball) is automatically removed.
  - setInvaders() - this method creates the invaders Blocks, pairs each one of them with the level's hitListeners (the BlockRemover and the ScoreTrackingListener), creates from them the invaders object and adds it to the game.
  - playOneTurn() - creates the spaceship, runs countdown and then the level, removes the remained bullets and the spaceship.

- doOneFrame() - draw all the sprites and notifies them the time passed. Checks if there are any invaders left, if it is time for them to shoot, if the player can shoot and tries to, and if the invaders hit the shield. Takes care of all of the above.

- Invaders - this class implements the Sprite interface, and it's main privates are:
  - private Block[5][10] matrix - which is mostly used to find out if an invader (Block) exists in a certain location.
  - private ArrayList<Block> blocks - all the invaders are simple blocks.

  The main methods of the invaders are:

  - timePassed(double dt) - moves the invaders and changes the private timeToShoot if it is time.
  - move(double dt) - moves the invaders. (detailed description below)
  - removeInvader(Block toRemove) - removes the Block from the list, and turns null the invader's location in the matrix.
  - Point nextShootingPoint() - returns the location for a bullet that will be shot from the lowest alien from a random column.
  - setMatrix() - set the remaining invaders in their places in the upper left corner of the screen, and update their speed to the original speed of the level.
- Spaceship - I took the Paddle class and changed it to be Spaceship, by adding the privates boolean wasHit, long lastShot and boolean okToShoot, and the next methods:
  - In the function "timePassed(double dt)" I added the update of okToShoot, according to lastShot's state.
  - updateThatJustShot() - is called from outside. Updates lastShot and okToShoot.

Implementations description:

- the Aliens formation - first, I kept a private boolean called moveRight. Every time the aliens needed to move, their direction was according to this boolean. Second, I created a method called numberOfColumns(), that calculates the current number of invaders columns that needs to be considered, and that's how the method "move()" knows if the formation is hitting the wall. If so, it calls "moveDown()" that moves the formation down and updates the moveRight boolean and the speed.
- The Shields - the shields are just three groups of very small blocks without stroke.
- Shots by Aliens - every time that doOneFrame() in GameLevel is called, it checks if isTimeToShoot() of its Invaders is true. If so, it creates a new bullet (Ball) in the location it gets from the nextShootingPoint() method of the invaders, adds it to the game and update the invaders that they shot using their updateThatJustShot() method. The Invaders knows when it is time to shoot, because every time their timePassed(double dt) method is called, the time is checked, and when it is right the private timeToShoot is being updated to "true".
- Shots by Player - every time that doOneFrame() in GameLevel is called, it checks if isOkToShoot() of its spaceship is true and if the space key is pressed. If so, it creates a new bullet (Ball) just above the spaceship, adds it to the game and update the spaceship that it shot using its updateThatJustShot() method. The spaceship knows it is ok to shoot because every time its timePassed(double dt) method is called, the time is checked and when it is right, the private okToShoot is being updated to "true".