**ROLL THE BALL**

**Problem Definition**

In this game, player should create a correct path for the ball from the start block to the end block by moving the randomly placed blocks in the correct order. To user to win the game, he/she should complete all the 5 levels. The difficulty of the game will increase in each new level.

**Implementation Details**

|  |  |
| --- | --- |
| **Blocks** | |
| **-**  **-**  **-**  **-**  **+**  **+**  **+** | id: int  type: String  property: String  img: ImageView |
| Blocks(int id, String type, String property)  Assign(): void  getter/setter methods |

* Blocks Class has several data fields, methods and getter/setter’s.
* Each blocks should have an id, a type, a property.
* In Assign method, program should sets block’s image into img variable.

|  |  |
| --- | --- |
| **Game** | |
| #  #  #  #  #  #  #  #  #  #  #  #  #  #  # | levelNumber: static int  winCounter: static int  moveCounter: static int  totalCounter: static int  winChecker: static boolean  counter: static Text  warning: static Text  levels: static ComboBox  blockList: ArrayList<Blocks>  pane: GridPane  mainPane: BorderPane  menuPane: Pane  close: Pane  howToPane: Pane  restartPane: Pane |
| +  +  +  +  +  + | moveToRight(int index, int column, ArrayList<Blocks> blockList,MouseEvent event,GridPane pane): void  moveToDown(int index, int row, ArrayList<Blocks> blockList,MouseEvent event,GridPane pane): void  moveToUp(int index, int row, ArrayList<Blocks> blockList,MouseEvent event,GridPane pane): void  moveToLeft(int index, int column, ArrayList<Blocks> blockList,MouseEvent event,GridPane pane): void  Win(ArrayList<blocks> blockList): boolean  levelGenerator(int levelNumber,ArrayList<Blocks> blockList,GridPane pane, BorderPane mainPane): void |

Game class is the main class in our game.

LevelNumber variable stores the current level.

winCounter variable is a variable that controls which level is completed or not and stores value due to that.

moveCounter variable stores how many times player moved blocks for each level.

totalCounter variable stores how many times player moved blocks until the end of the game.

winChecker variable controls that level is completed or not.

counter is a text object that shows user how many times player moved blocks for each level.

warning is a text object that warns player if player tries to move on to the next level without passing the current level.

levels is a ComboBox object that allows user to choose level to play.

blockList is an ArrayList that holds our block objects. We use that arraylist for many place like generating levels. dragging blocks, check whether level is completed or not.

Pane is the pane that we use for setting blocks into right indexes.

mainPane is the pane that takes everything and put them into right place in the scene.

menuPane is the pane that shows user to main menu. Player can choose start game, how to play or exit game.

close is the pane that appears when user attempt to exit game.

howToPane is the pane that shows user details about game.

restartPane is the pane that appears when user complete all levels. It shows message due to totalCounter variable and allows user to play the game again.

moveToRight, moveToLeft, moveToDown, moveToUp methods invoked when user tries to drag blocks. These methods swaps the index blocks in the pane and also swaps them in the blockList with checking the MouseEvent.

Win method checks the properties of the objects in the blockList and controls the level is completed or not and returns a boolean value due to that.

levelGenerator method takes the input from a file due to current level and sets pane and create block objects and put that objects into blockList due to that input file. Everytime that method invoked, it clears pane and blockList than sets them with new values.

We created Blocks class according to the UML diagram. With this class we can arrenge and organize blocks due to inputs easily. Also we can control whether level is completed or not.

We have BorderPane named mainPane. We use this pane for showing everything to user in the primaryStage when user playing the game.

We created GridPane named pane. With that pane we arrange Blocks objects that we created in the scene. Also we put every Blocks objects inside of an blockList ArrayList in the correct order. For swapping blocks we used 2 events. Firstly we created MousePressedEvent for recognize indexes of Blocks objects that user pressed. Than we created MouseDraggedEvent inside of that event for recognizing index of the Blocks object that user want to swap. We have 4 different methods(moveToDown, moveToUp,moveToLeft,moveToRight) for controlling these kind of moves. In these method we checked properties of these objects. If these objects able to swap, we change these 2 objects indexes in the pane and also swap them in the blockList. In Win method we check their types and properties of every Blocks object in the arraylist for controlling whether level is completed or not.We invoked Win method in every MouseDraggedEvent. If this method returns true, we checked and set the path of animation for that level than play the animation. Also when Win method returns true on MouseEvent we set the value of winChecker variable as true. Thus we prevent user from doing MouseDraggedEvent with an if statement.

For generating levels we created levelGenerator method.This method takes integer levelNumber , ArrayList blockList, GridPane pane and BorderPane mainPane. When that method invoked it clears pane and blockList and remove them from mainPane than it generates them again acording to levelNumber and put them into mainPane.

To generate the ball we created circle object inside of levelGenerator method and add it into mainPane.

We created ComboBox named levels to allow player to play the level that he/she wants to play. We use setOnAction event for this ComboBox to do that. According to selected level in the ComboBox, we invoked levelGenerator method.

We have winCounter variable for controling the completed levels. If current level is completed for the first time we increment the value of winCounter by controlling Win method.

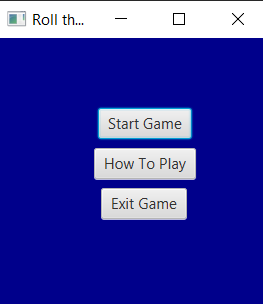
By using winCounter we prevent user from passing next level without complete the current level.

We use moveCounter and totalCounter to count the swaps that player does. When MouseDragEvent invoked, these variables were incremented. moveCounter is for counting the swaps for current level and resets in each level. Game can be completed with at least 27 swaps. According to totalCounter, game shows a message when player finished the game.

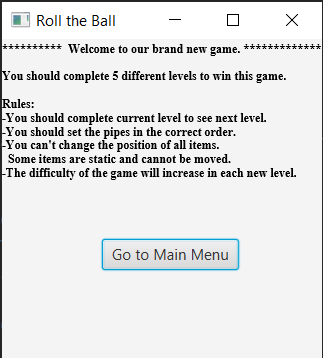
We create menuPane for make main menu in our game. In this menu user can choose start to play, look How To Play for take information about game or exit from game.

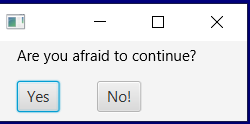
We create restartPane for showing message and asks user to play the game again or not. It also shows message according the total movement on the game. There are several messages due to players performation. User can try to play again for better results in the end.

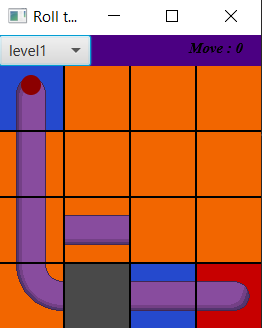
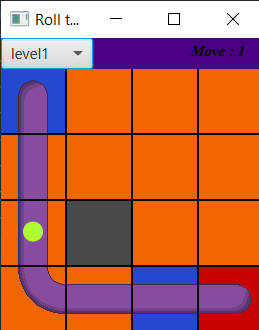
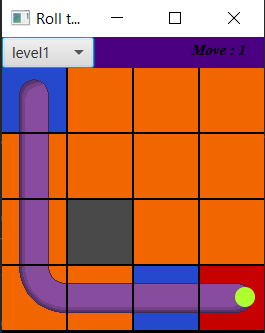
**Test Cases**

****

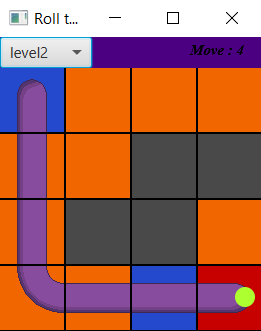
This is a screenshot from our main menu. Player can press Start Game button to play the game, How To Play button to get information about game or Exit Game to exit from game.

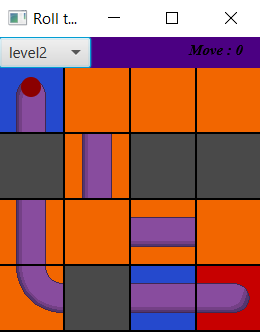
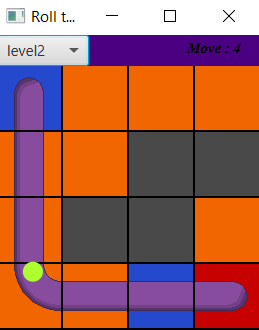
This is a screenshot from our How To Play scene. Player can press Go to Main Menu button to back to the menu.

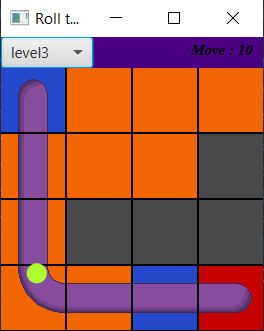
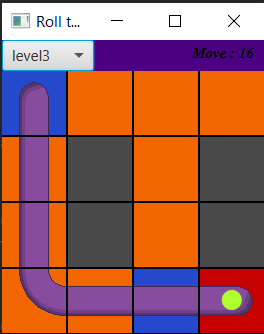
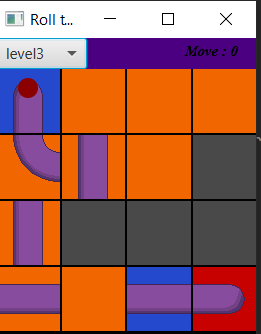
This screenshot is taken from our exit stage. Player can choose whether continue to play or exit from game. This stage opens when player press Exit Game button or Trying to close game.



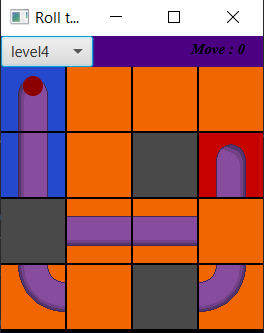
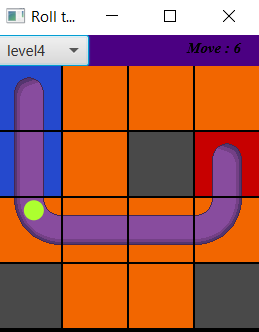
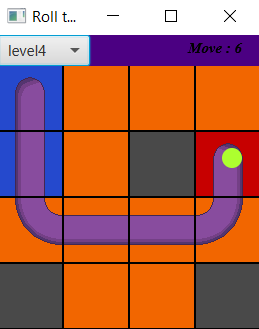
These images above are from first level of the game. First image is the begining of the level. When user complete the level, color of ball turns to green from red and animation plays. User can choose next level from box on the top left. User can also see how many moves did he/she make on the current level from moves text on the top right.



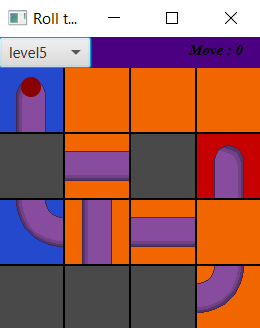
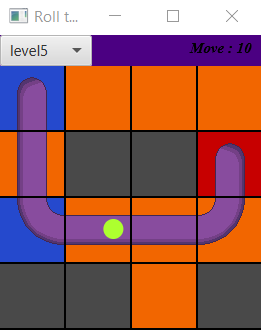
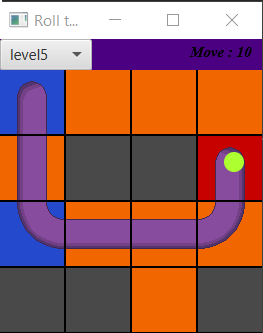
These images above are from level 2.

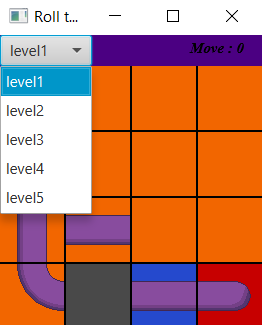


These images above are from level 3.

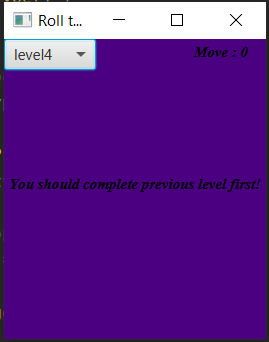


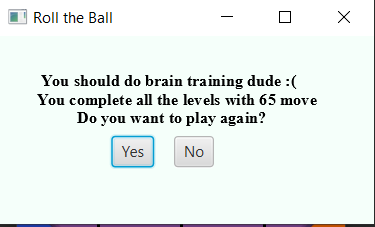
These images above are from level 4.

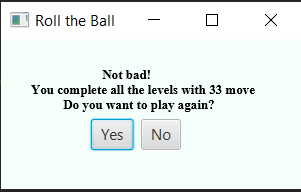
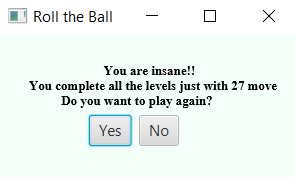
These images above are from level 5.



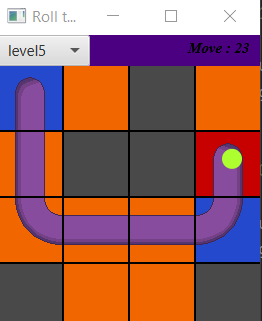
Player can change the current level from this ComboBox.

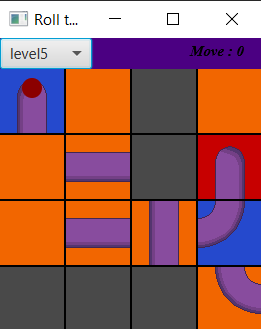
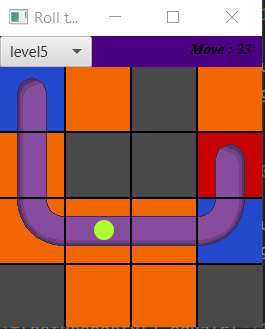
This message displays when player tries to play following levels without completing previous levels.





These messages appear when user completed all levels. This game can be completed with at least 27 move. According the total moves that player does, These messages appear on the scene and criticises the player.





These images above are from level that we created for demo phase. We changed the input of level5 text file to our values for game. With that we checked our game is playable for various inputs.