MARMARA UNIVERSITY

FACULTY OF ENGINEERING

Computer Engineering

CSE1142

Computer Programming II, Spring 2020

TERM PROJECT

PIPE GAME

NAME	STUDENT NUMBER
Yasin Tarakçı	150118055
Yusuf Taha Atalay	150119040

Date Submitted: May 10, 2020

PROBLEM DEFINITION

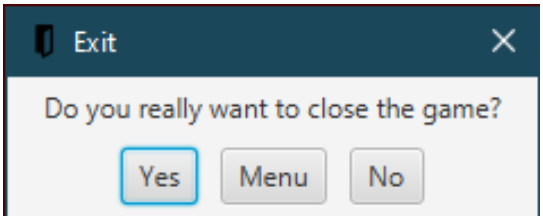
Our game is a simple pipe game. The player have to build the correct path with pipes in order to make ball move from start to end. The game is tile based the player can move available tiles to the free spaces. Our game has five levels. Each level have different tile sequence and different path to complete level.

In this game we have 4 different types of tiles. The first two types are starter tile and end tile. These can be vertical or horizontal and these tiles can't move. The other one is empty tile. This tile has two subtypye, one of them is static and the other one can move to free tile. The last one is pipe tile. There is two subtypes. The first subtype is straight pipe and this can be vertical or horizontal. Other subtype is curved pipe. These subtypes also have two kind, one of them is static and the other one is movable. These pipe tiles make the ball move in arc path.

IMPLEMENTATION DETAILS

Confirm Box	
-	answer : int
+	menuButton : Button
+	**display(title: String , message: String): int**

- in display method its create a scene with given message and title parameters.Here is an example output;



Tiles	
-	tile_image : ImageView
-	tile_id: int
-	type: String
-	property: String
+	**Tiles(tile: int, type: String, property: String)**
+	**toString() : String**
+	**getter / setter methods**

- Tiles() method assigns corresponding tile image to tile name.
- There are setter/getter methods.

Main	
-	pathOne1, pathOne2, pathOne3: PathTransition
-	pathTwo1, pathTwo2, pathTwo3, pathTwo4, pathTwo5 : PathTransition
-	tiles_level1, tiles_level2, tiles_level3, tiles_level4, tiles_level5: Tiles[]
-	level1, level2, level3, level4, level5: String
-	current_level: String
-	step_counter_label: Label
-	total_counter_label: Label
-	borderPane: BorderPane
-	paneForStepCounters: BorderPane
-	start_pane: Pane
-	hBoxForButtons: HBox
-	level1_button, level2_button, level3_button, level4_button, level5_button : Button
-	start_button: Button
-	credit_button: Button
-	quit_button: Button
-	step_counter_text: Text
-	total_counter_text: Text
-	win_text: Text
-	steps_counter: int
-	total_counter: int
+	**start(primaryStage: Stage): void**
+	**closeProgram(): void**
+	**button_handler(level: String, tiles_level: Tiles[]): void**
+	**path_maker(level: String): void**
+	**animation_paly(level: String): void**
+	**move_cell(cell_index: int, gridPane: GridPane, tiles: Tiles[]): void**
+	**step_controller(cell_index: int, tiles: Tiles[], next: Tiles): void**
+	**map_maker(level: String, tiles: Tiles[]): GridPane**
+	**isWon(tiles: Tiles[]): boolean**
+	**can_move_right(cell_index: int, tiles: Tiles[]): boolean**
+	**can_move_left(cell_index: int, tiles: Tiles[]): boolean**
+	**can_move_up(cell_index: int, tiles: Tiles[]): boolean**
+	**can_move_down(cell_index: int, tiles: Tiles[]): boolean**
+	**main(args String[]): void**

- start() method creates the overall look of the game with the help of other methods.
- cloaseProgram() method first asks the user if the player is sure to quit via confirmbox if yes it closes the program.
- button_handler() method creates and updates the level and step counters.
- path_maker() method creates path for each level according to the input files.
- animation_play() method plays the ball movement animation if the correct path has occured.
- move_cell() method allows user to move suitable cells to free spaces.
- step_controller() method controls if the movement player want to do is allowed and also un-fades buttons if current level has completed.
- map_maker() method creates visual map according to the input files.
- isWon() checks the map after every movement to see if required path has maded if so then returns true.

- can_move_left() method checks if the selected tile has is movable and if so can it move to the left.
- can_move_right() method checks if the selected tile has is movable and if so can it move to the right.
- can_move_up() method checks if the selected tile has is movable and if so can it move to the up.
- can_move_down() method checks if the selected tile has is movable and if so can it move to the down.

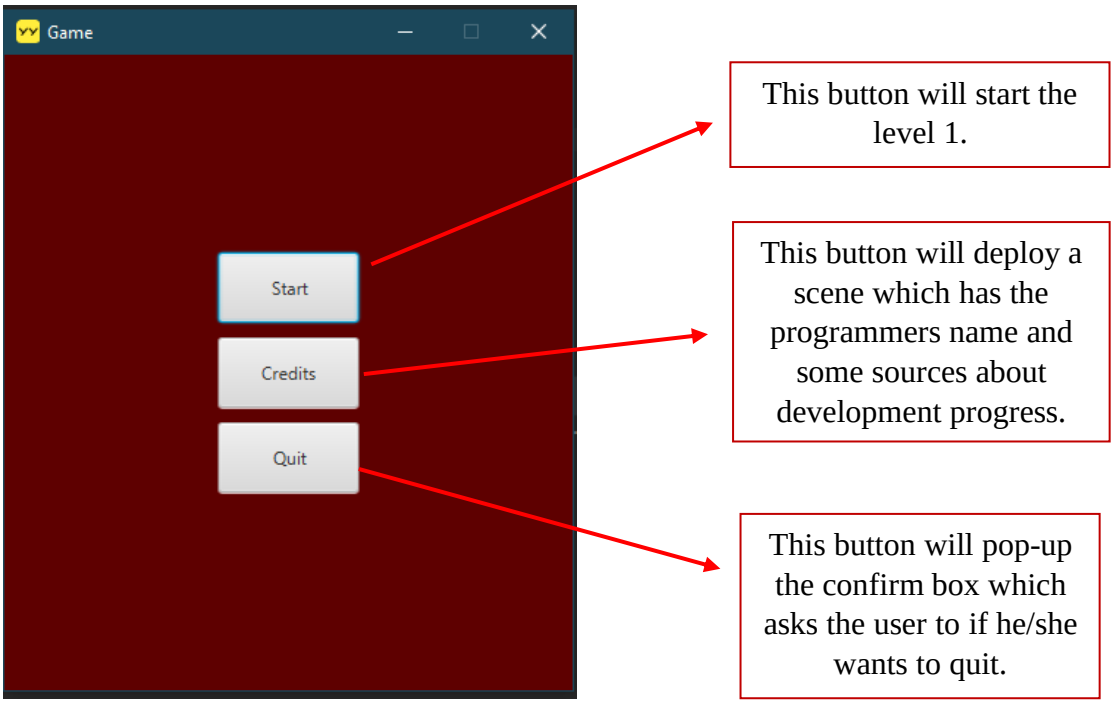
At the beginning we had a problem for moving the tiles. After making some research about mouse events we figure it out to use mouse events, in order to have more functionality on tiles we decided to create tiles class and stack all the tiles in one array.

In our game we these functionalities as extra:

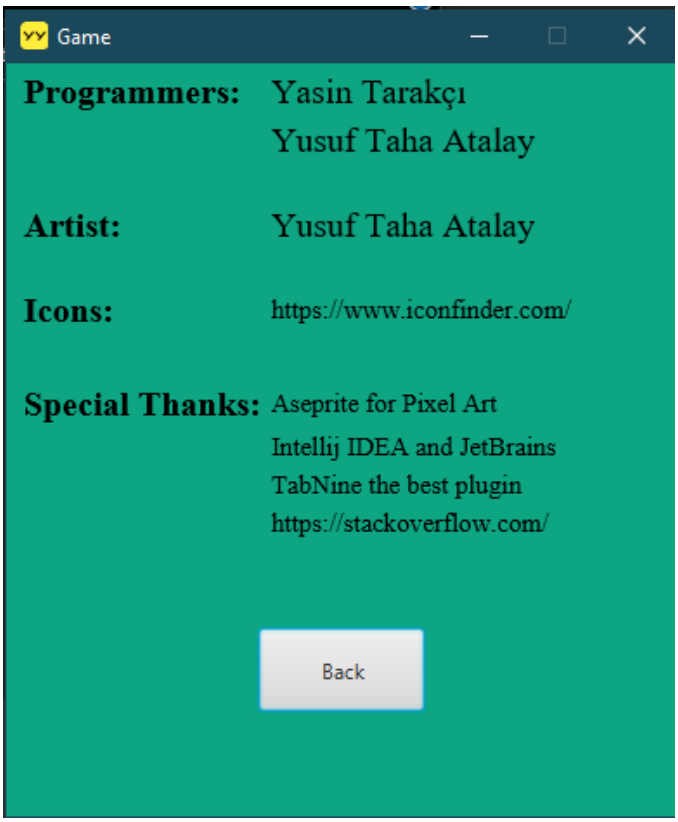
- Start Screen: When program first opened the player sees this scene with 3 buttons. First one is Start button and this button starts the game. The scond one is credits button. This button takes user to another scene which has some intormation about programmers and the sources they use. Last one is quit button. This button pops a confirm box.
- Confirm Box: When the user wants to close the game there appears a window with yes, no and menu button.
- Level Buttons: These buttons are faded-out and non-clickable except level 1 button when the game starts. As the user completes the current level next level's button will fade-in and become clickable.
- And our game has an icon.

TEST CASES

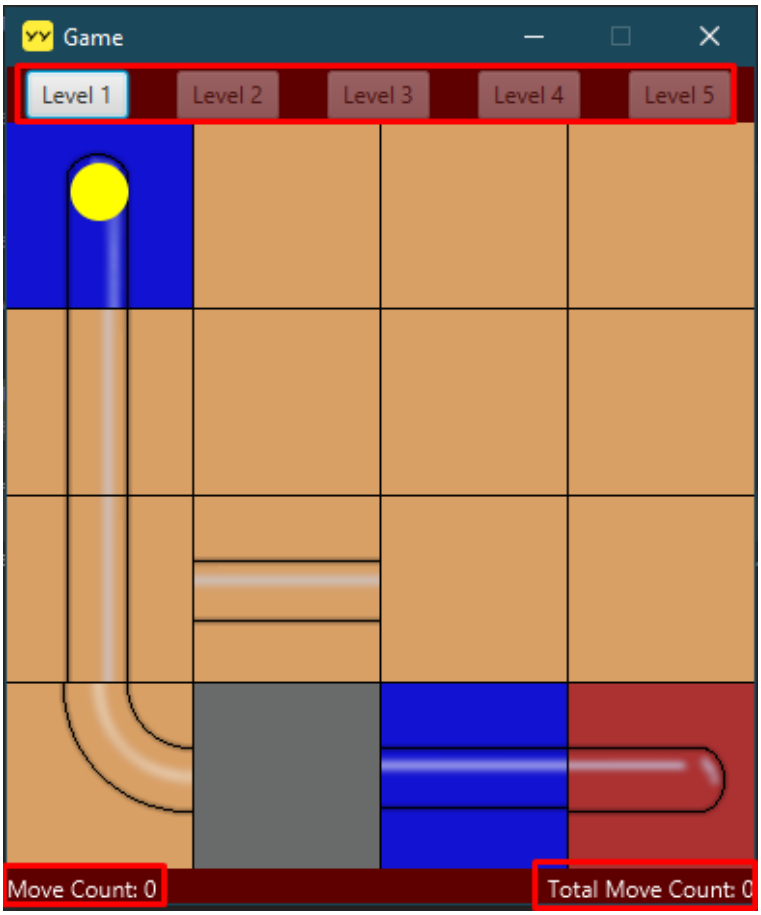
When the program starts, the first thing the user would see the start scene:



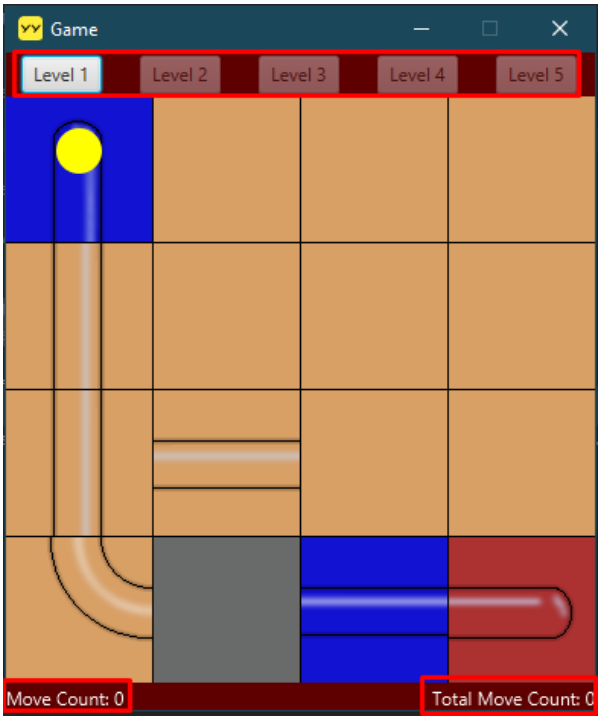
When the credits button clicked the following scene will shown:

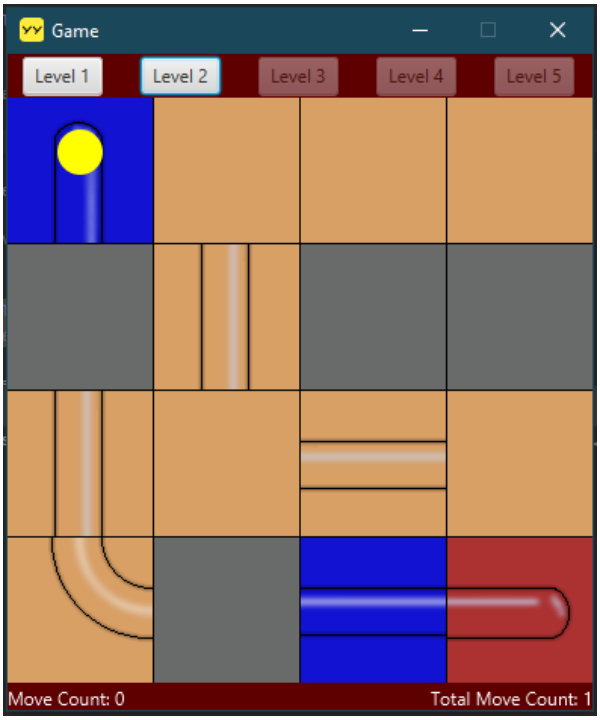


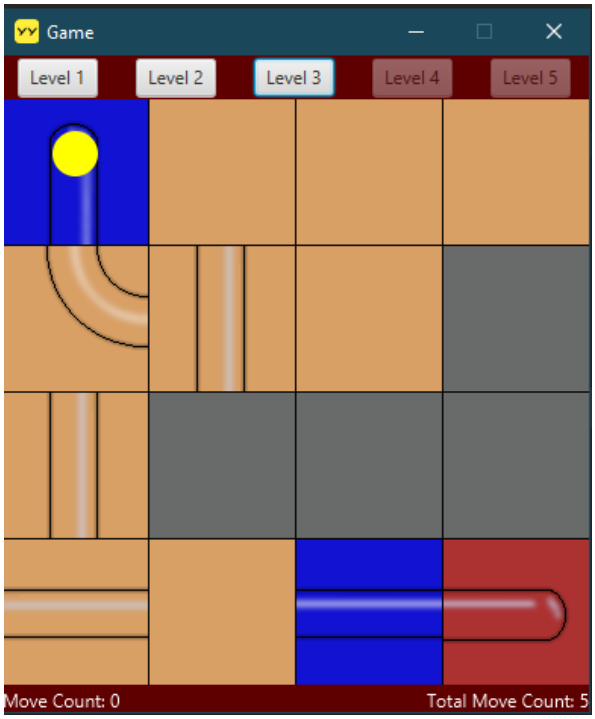
When the start button clicked level 1 will started and move counter counts every step in each level when new level starts move counter get reseted:

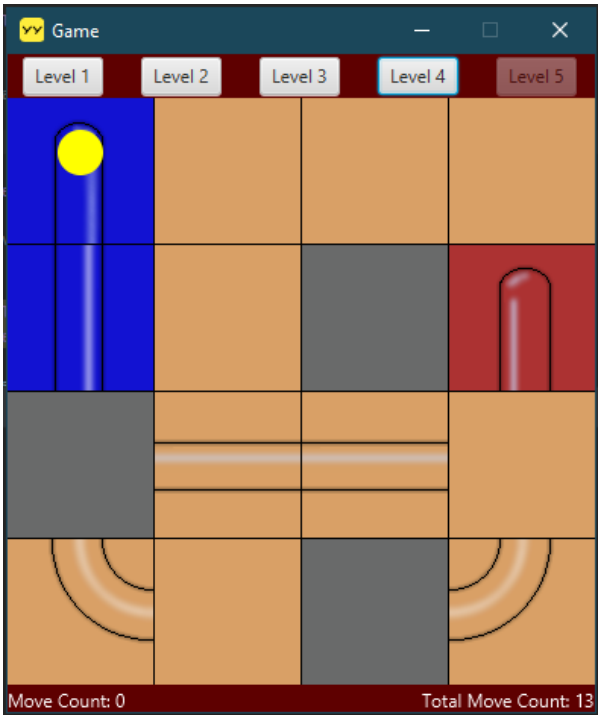


When the current level is completed the next level's button will enabled:

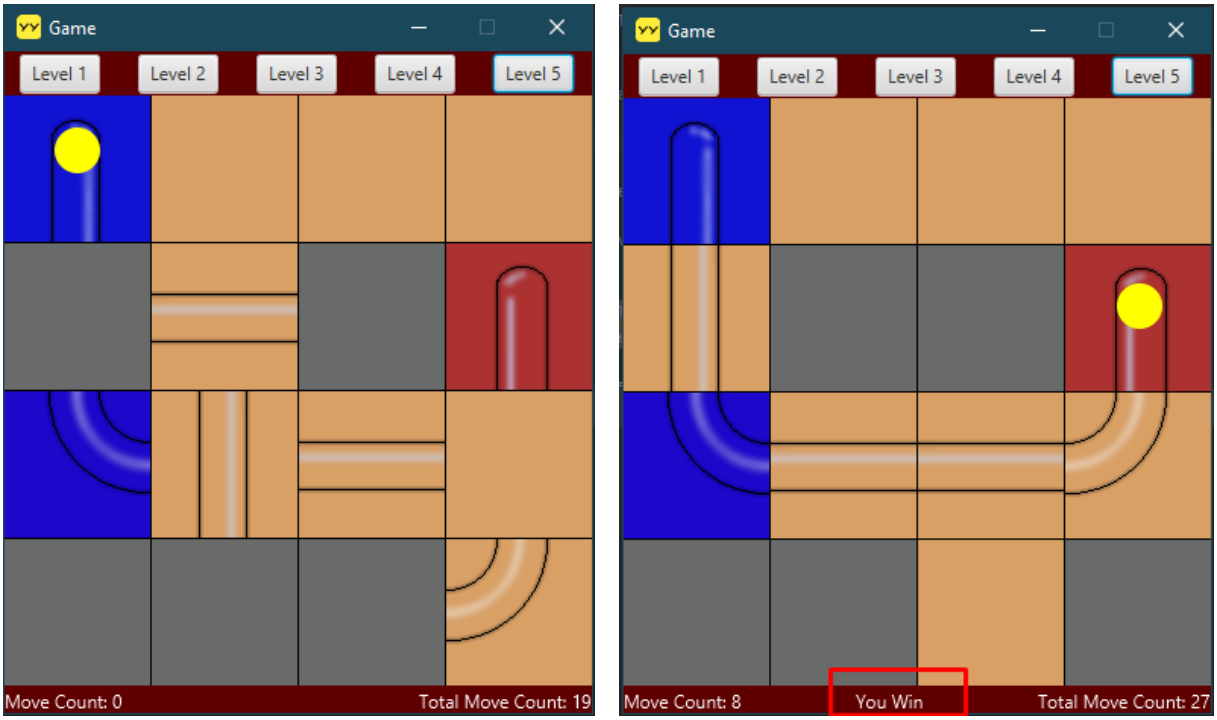




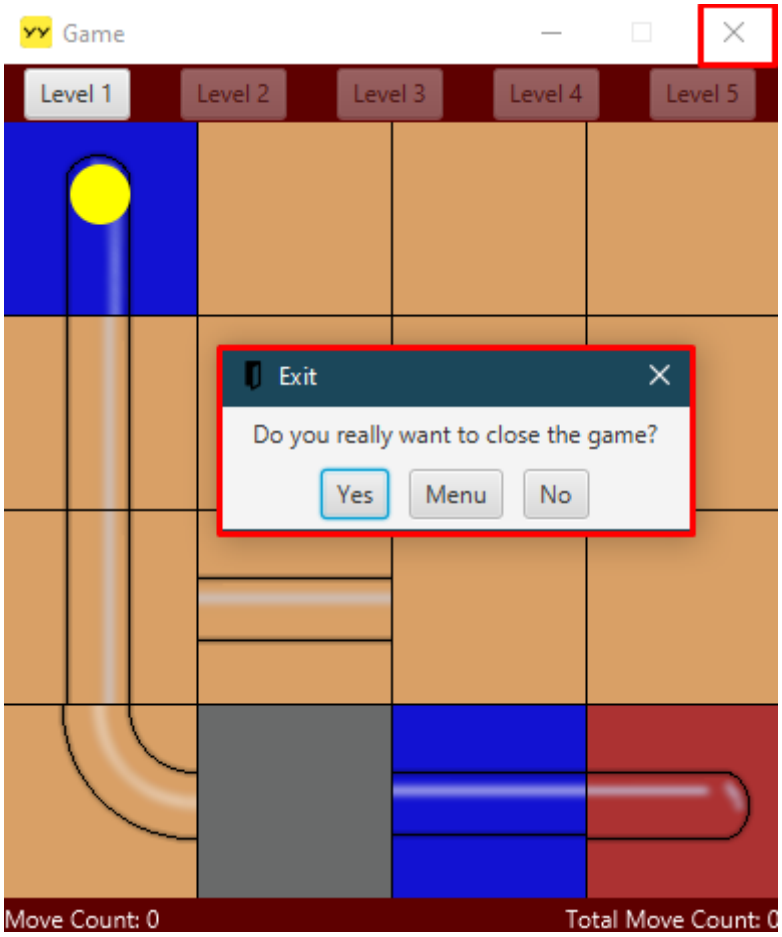




When the level is completed the ‘You Win’ text will be shown between two move counts:



When the quit button or the cross button on the top right corner is clicked the following scene will shown:



The following screenshots will demonstrate mouse drag events on tiles: