

Create the Game 2048 with Java 8 and JavaFX

By developing the famous game 2048 with JavaFX and Java 8 in this hands-on lab session, you will encounter several new language features such as Stream API, lambda expressions, and new util classes and methods. You will also learn basic concepts of JavaFX 2-D animations, parallel processing, UI design, CSS styling, and more.

Project Base

Open the repo and have a look at the project:

 <https://github.com/jperedadnr/Game2048H0L.git>

The starting project is a JavaFX application with the following Java classes:

- Game2048
- GameManager
- Board
- Tile
- Location
- Direction
- GridOperator

And the following resources to style the visual:

- game.css
- ClearSans-Bold.ttf

Some classes and methods are lacking implementations. Follow this guide step by step to fulfill the missing code. Complete the methods and have a fully operative version of the game 2048 FX. Basically you'll just have to copy and paste the snippets of code available on each step.

Please follow carefully every step. Try to build and run the application every now and then to check you didn't miss any step and everything is in place. Feel free to ask any questions along the way.

STEP 0. Clone the repository

Open NetBeans 8.0 and under **Team -> Git** select **Clone** and type this URL:

 `https://github.com/jperedadnr/Game2048Empty.git`

- Accept the default destination folder or choose one
- Select branch and click Next/Finish
- Wait till the local copy is created, and the project is opened in NetBeans
- Select Build and Run to test the (empty) application

Now follow these steps

INDEX

Starting with the basis

1. Add GameManager into Game2048
2. Add a Board into GameManager
3. Create a score to visualize points
4. Define rectangle corner border size
5. Draw a grid of rectangles in the Board
6. Add Tiles into the grid
7. Create a random Tile
8. Layout the tile at its location
9. Let's start the Game!

Styling

10. CSS styling

Basic movements

11. Connect arrow keys with Direction
12. Moving one tile
13. Moving tiles on the board
14. Listening to arrow keys
15. Initializing the game
16. Random tiles, random locations

17. How far can a tile go?

Animation

18. A new approach to move the tiles

19. Animating one tile movement

20. Animating all the tiles together

21. Find a random available location

22. Adding and animating new tiles

23. Playing with new cells on the board

Java 8 Power

24. Introducing the GridOperator

25. Traversing the grid properly

26. Sorting lists before moving tiles

27. Let's start merging tiles

28. Animating the merged tiles

29. Performing tiles merging on the move

30. Cleaning merged tiles

31. Fixing undesired effects

Score

32. Adding points

33. Counting points

34. Centering the points

35. Animating the points earned

36. Looking for pairs of mergeable tiles

37. Checking for Game Over

38. Checking for the winning tile

Overlay

39. Setting options buttons

40. Setting up the Overlay

41. Listening to game won/over

42. Want to try again?

Optional

43. Optional tiles

44. Finding mergeable (optionally null) tiles

45. Traversing the grid with Optional

Not included

46. Extra features

STEP 1. Add GameManager into Game2048

Create an instance of GameManager in Game2048 class and add it to the root object (StackPane).

SOLUTION CODE

- *Class*: Game2048
- *Method*: start
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Game2048.java#L34-35>)
- Copy and paste the following code snippet:

```
gameManager = new GameManager();  
root.getChildren().add(gameManager);
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 2. Add a Board into GameManager

Create an instance of Board in GameManager class and add it to its list of children.

SOLUTION CODE

- *Class*: GameManager
- *Method*: constructor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L50-51>)
- Copy and paste the following code snippet:

```
board = new Board();  
getChildren().add(board);
```

STEP 3. Create a score to visualize points

Create nodes for `hTop` in `createScore()`, with these steps:

- Create a `Label` named `lblTitle` with text "2048" and other named `lblSubtitle` with text "FX"
- Create an empty `HBox` named `hFill` and set its horizontal grow priority to grow always
- Create a `VBox` named `vScores` and an `HBox` named `hScores` with spacing 5
- Create a `Label` named `lblTit` with text "SCORE", and add it to `vScore`, as well as `lblScore`
- Create a `VBox` named `vRecord` and a `Label` named `lblTitBest` with text "BEST", and add it to `vRecord`, as well as `lblBest`
- Add `vScore` and `vRecord` to `hScores`, create a `VBox` named `vFill` with vertical grow priority to grow always, and add `hScores` and `vFill` to `vScores`
- Finally, add `lblTitle`, `lblSubtitle`, `hFill` and `vScores` to `hTop`

SOLUTION CODE

- *Class*: `Board`
- *Method*: `createScore()`
- *preview*
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Board.java#L77-98>)
- Copy and paste the following code snippet:

```
 createScore() {  
    Label lblTitle = new Label("2048");  
    Label lblSubtitle = new Label("FX");  
  
    HBox hFill = new HBox();  
    HBox.setHgrow(hFill, Priority.ALWAYS);  
  
    VBox vScores = new VBox();  
    HBox hScores = new HBox(5);  
  
    Label lblTit = new Label("SCORE");  
    vScore.getChildren().addAll(lblTit, lblScore);  
  
    VBox vRecord = new VBox(0);  
    Label lblTitBest = new Label("BEST");  
    vRecord.getChildren().addAll(lblTitBest, lblBest);  
  
    hScores.getChildren().addAll(vScore, vRecord);  
    VBox vFill = new VBox();  
    VBox.setVgrow(vFill, Priority.ALWAYS);
```

```

        vScores.getChildren().addAll(hScores, vFill);

        hTop.getChildren().addAll(lblTitle, lblSubtitle, hFill, vScores);
    }

```

Call the method `createScore()` from the constructor of `Board`

SOLUTION CODE

- *Class*: `Board`
- *Method*: constructor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ho1/game2048/Board.java#L66>)
- Copy and paste the following code snippet:

```
createScore();
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 4. Define rectangle corner border size

Create a rectangle located in corners `i*cell_size, j*cell_size`, of size `cell_sizeXcell_size`, filled with white color, and with border grey

SOLUTION CODE

- *Class*: `Board`
- *Method*: `createCell`
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ho1/game2048/Board.java#L170-172>)
- Copy and paste the following code snippet:

```

cell = new Rectangle(i * CELL_SIZE, j * CELL_SIZE, CELL_SIZE, CELL_SIZE);
cell.setFill(Color.WHITE);
cell.setStroke(Color.GREY);

```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 5. Draw a grid of rectangles in the Board

Add 4x4 cells to `gridGroup`, by calling `createCell` method for every cell

SOLUTION CODE

- *Class*: `Board`

- *Method:* createGrid

- preview

(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Board.java#L185-189>)

- Copy and paste the following code snippet:

```
for(int i=0; i<4; i++){
    for(int j=0; j<4; j++){
        gridGroup.getChildren().add(createCell(i, j));
    }
}
```

Call method createGrid() from constructor of class Board

SOLUTION CODE

- *Class:* Board

- *Method:* constructor

- preview

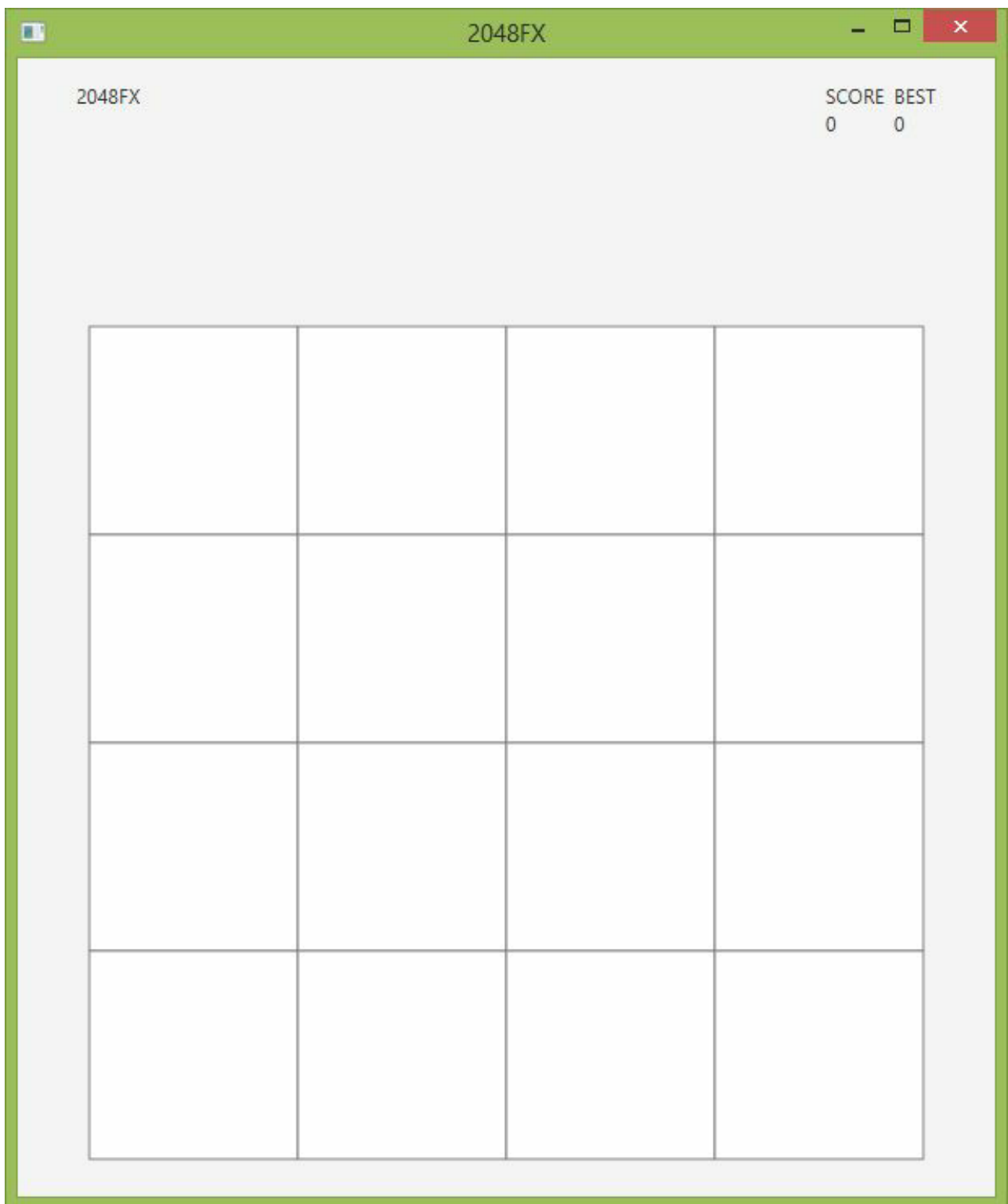
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Board.java#L66>)

- Copy and paste the following code snippet:

```
createGrid();
```

Screenshot after Step 5

Run the project to see the application after completing the first 5 steps



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 6. Add Tiles into the Grid

- Set the size of the tile to `Board.CELL_SIZE-13` to account for the stroke width
- For now, style the tile background with `#c9c9c9` color
- Set the tile alignment centered
- Assign the value from the argument to `value`

- Finally, set the text with this value, and initialized `merged` as `false`.

SOLUTION CODE

- *Class*: `Tile`
- *Method*: `private constructor`
- `preview`
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Tile.java#L32-46>)
- Copy and paste the following code snippet: ```java final int squareSize = Board.CELL_SIZE - 13; setMinSize(squareSize, squareSize); setMaxSize(squareSize, squareSize); setPrefSize(squareSize, squareSize); setStyle("-fx-background-color: #c9c9c9;"); setAlignment(Pos.CENTER);`

`this.value = value; this.merged = false; setText(Integer.toString(value));`

```

Back to [Index][I0]
***
## STEP 7. Create a random Tile
Create a new `Tile` instance, which value being **2** with a 90% possibility or *
*4** with the 10% remaining

### SOLUTION CODE
* *Class*: `Tile`
* *Method*: `newRandomTile`
* [preview][7]
* Copy and paste the following code snippet:
``java
return newTile(new Random().nextDouble() < 0.9 ? 2 : 4);

```

Back to Index (<https://github.com/jperedadnr/Game2048HOL#index>)

STEP 8. Layout the tile at its location

Set the tile layout by getting from its location the position at the center of the cell and subtracting half of the tile dimensions

SOLUTION CODE

- *Class*: `Board`
- *Method*: `moveTile`
- `preview`
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L234-238>)
- Copy and paste the following code snippet:

```

double layoutX = tile.getLocation().getLayoutX(CELL_SIZE) - (tile.getMinWid
| th() /

```

```

2);
double layoutY = tile.getLocation().getLayoutY(CELL_SIZE) - (tile.getMinHeight()
/ 2);
tile.setLayoutX(layoutX);
tile.setLayoutY(layoutY);

```

Add a call to `moveTile` when a tile is added

SOLUTION CODE

- *Class*: Board
 - *Method*: `addTile`
 - preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Board.java#L226>)
 - Copy and paste the following code snippet:


```

moveTile(tile, tile.getLocation());

```
- [Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)
-

STEP 9. Let's start the Game!

Add a random tile a location of your choosing to the board

SOLUTION CODE

- *Class*: GameManager
- *Method*: `startGame`
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L107-111>)
- Copy and paste the following code snippet:


```

Tile tile0 = Tile.newRandomTile();
tile0.setLocation(new Location(1,2));
board.addTile(tile0);

```

Add a call to `startGame` in the `GameManager` constructor

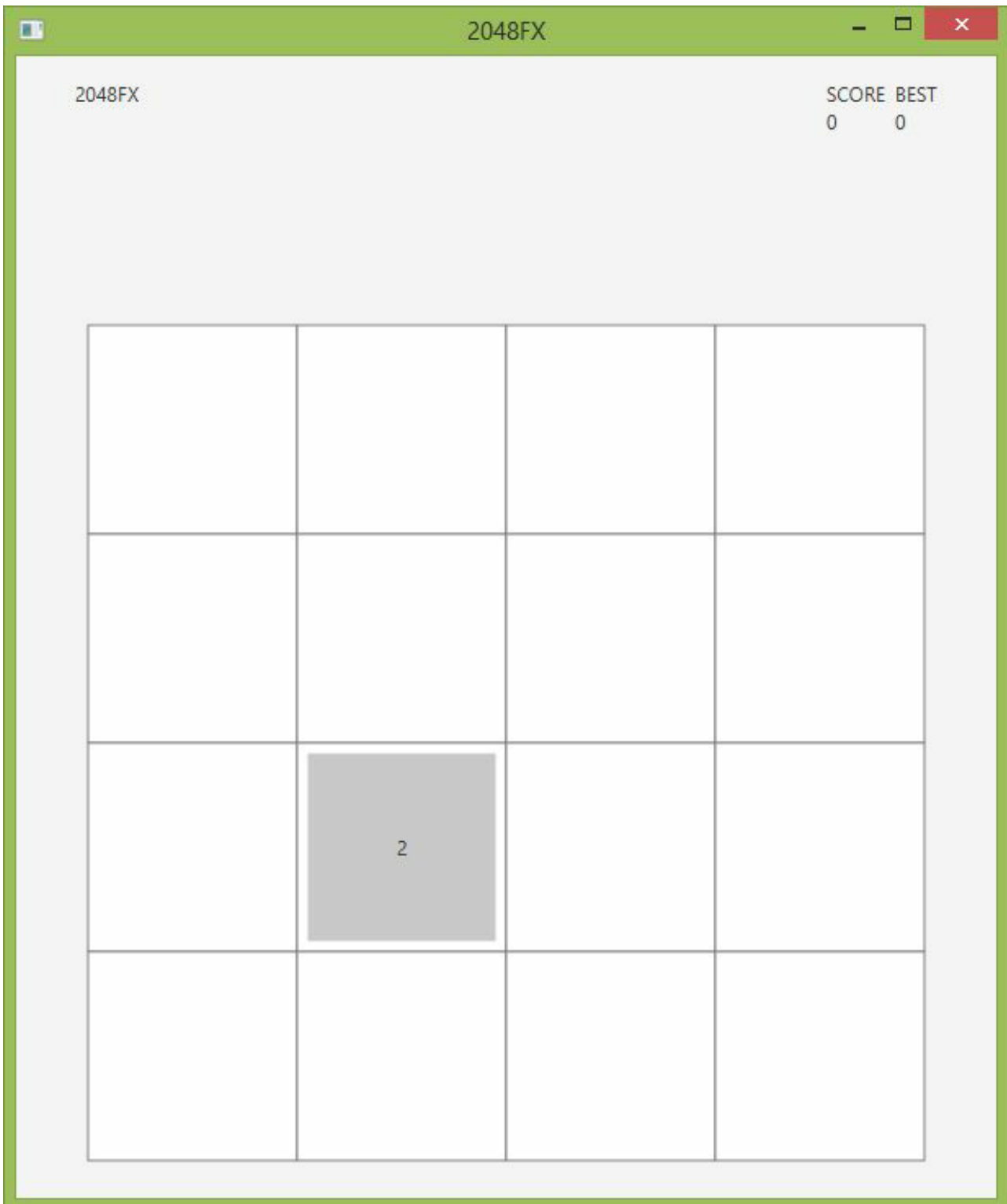
SOLUTION CODE

- *Class*: GameManager
- *Method*: constructor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L69>)
- Copy and paste the following code snippet:

```
startGame();
```

Screenshot after #9

Run the project to see the application after completing the first 9 steps



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 10. CSS styling

Load the font *ClearSans-Bold.ttf* at the beginning of the application

SOLUTION CODE

- *Class*: Game2048
- *Method*: init
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/hol/game2048/Game2048.java#L24>)
- Copy and paste the following code snippet:

```
Font.loadFont(Game2048.class.getResource("ClearSans-Bold.ttf").toExternalForm(), 10.0);
```

Enable css styling in the application by loading the *game.css* file. Apply "game-root" selector to the root

SOLUTION CODE

- *Class*: Game2048
- *Method*: start
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/hol/game2048/Game2048.java#L40-41>)
- Copy and paste the following code snippet:

```
scene.getStylesheets().add(Game2048.class.getResource("game.css").toExternalForm());  
root.getStyleClass().addAll("game-root");
```

Apply the styles to nodes in the hTop container:

- lblTitle: "game-label", "game-title"
- lblSubtitle: "game-label", "game-subtitle"
- vScore and vRecord: "game-vbox"
- lblScore: "game-label", "game-score"
- lblTit: "game-label", "game-titScore"
- lblTitBest: "game-label", "game-titScore"
- lblBest: "game-label", "game-score"

SOLUTION CODE

- *Class*: Board
- *Method*: createScore
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/hol/game2048/Board.java#L102-109>)

- Copy and paste the following code snippet:

```
lblTitle.getStyleClass().addAll("game-label", "game-title");
lblSubtitle.getStyleClass().addAll("game-label", "game-subtitle");
vScore.getStyleClass().add("game-vbox");
lblTit.getStyleClass().addAll("game-label", "game-titScore");
lblScore.getStyleClass().addAll("game-label", "game-score");
vRecord.getStyleClass().add("game-vbox");
lblTitBest.getStyleClass().addAll("game-label", "game-titScore");
lblBest.getStyleClass().addAll("game-label", "game-score");
```

Adjust arc size in cells to `cell_size/6` and apply "game-grid-cell" style

SOLUTION CODE

- *Class*: Board
- *Method*: createCell
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L176-178>)
- Copy and paste the following code snippet:

```
cell.setArcHeight(CELL_SIZE/6d);
cell.setArcWidth(CELL_SIZE/6d);
cell.getStyleClass().add("game-grid-cell");
```

Apply "game-grid" to gridGroup and "game-backGrid" to hBottom

SOLUTION CODE

- *Class*: Board
- *Method*: createGrid
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L201-202>)
- Copy and paste the following code snippet:

```
gridGroup.getStyleClass().add("game-grid");
hBottom.getStyleClass().add("game-backGrid");
```

Apply to tiles the styles "game-label" and "game-tile-"+value, and remove the previously assigned style.

SOLUTION CODE

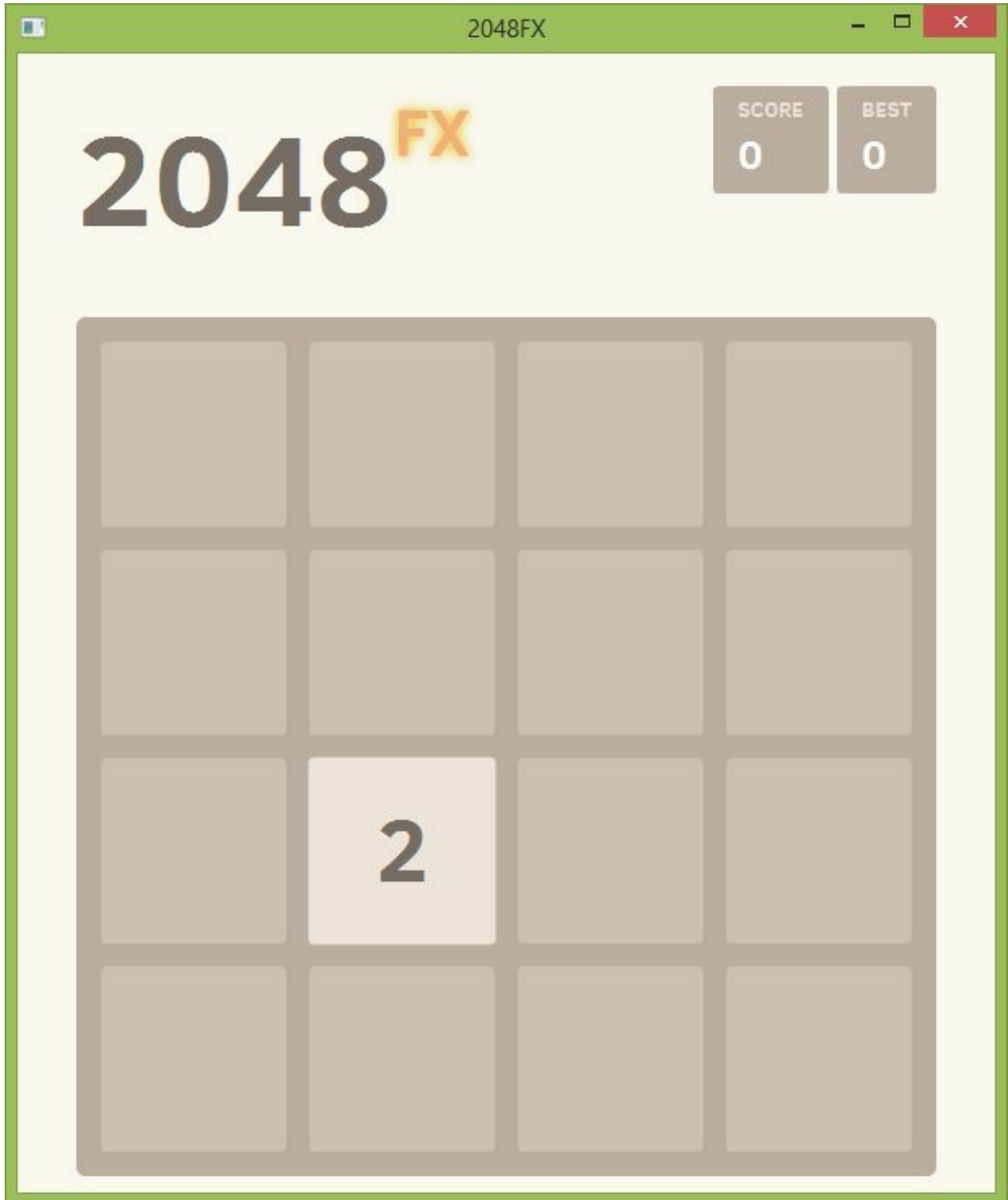
- *Class*: Tile
- *Method*: private constructor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Tile.java#L40>)

- Copy and paste the following code snippet:

```
getStyleClass().addAll("game-label", "game-tile-" + value);
```

Screenshot after #10

Run the project to see the application after completing the first 10 steps



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 11. Connect arrow keys with Direction

Return the enum constant of the type with the specified name from a KeyCode

SOLUTION CODE

- *Class:* Direction
- *Method:* valueFor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Direction.java#L37>)
- Copy and paste the following code snippet:

```
 return valueOf(keyCode.name());
```


[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 12. Moving one tile

Return a new Location based on the actual and an offset in the specified direction

SOLUTION CODE

- *Class:* Location
- *Method:* offset
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Location.java#L64>)
- Copy and paste the following code snippet:

```
 return new Location(x + direction.getX(), y + direction.getY());
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 13. Moving tiles on the board

- Get a list of tiles in the `gridGroup` and then remove all the list from the `gridGroup`
- Create new tiles based in the old ones, applying an offset in the specified direction to their current location, checking if the new location is valid and it doesn't contain another tile. Otherwise keep the previous location. Then add them to the `gridGroup`

SOLUTION CODE

- *Class:* GameManager
- *Method:* move
- preview

(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L157-170>)

- Copy and paste the following code snippet:

```
List<Tile> tiles=board.getGridGroup().getChildren().stream()
    .filter(g->g instanceof Tile).map(t->(Tile)t)
    .collect(Collectors.toList());
board.getGridGroup().getChildren().removeAll(tiles);
tiles.forEach(t->{
    Tile newTile = Tile.newTile(t.getValue());
    final Location newLoc=t.getLocation().offset(direction);
    if(newLoc.isValidFor() &&
        !tiles.stream().filter(t2->t2.getLocation().equals(newLoc)).findAny()
        .isPresent()){
        newTile.setLocation(newLoc);
    } else {
        newTile.setLocation(t.getLocation());
    }
    board.addTile(newTile);
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 14. Listening to arrow keys

Add a listener to the scene for keys pressed, then get the key code, check if it is an arrow key. In such case get the direction for that arrow and move tile

SOLUTION CODE

- *Class*: Game2048
- *Method*: start
- preview

(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Game2048.java#L45-51>)

- Copy and paste the following code snippet:

```
scene.setOnKeyPressed(ke -> {
    KeyCode keyCode = ke.getCode();
    if(keyCode.isArrowKey()){
        Direction dir = Direction.valueFor(keyCode);
        gameManager.move(dir);
    }
});
```

Screenshot after #14

Run the project to see the application after completing the first 14 steps. Press the right arrow

and check the tile from Screenshot #10

(<https://raw.githubusercontent.com/jperedadnr/Game2048HOL/master/src/doc/screenshot-Step10.jpg>) moves one cell to the right



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 15. Initializing the game

Clear the gameGrid map and the locations list, and then initialize both with all 4x4 valid

locations, and null tiles

SOLUTION CODE

- *Class*: GameManager
- *Method*: initializeGameGrid
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L79-89>)
- Copy and paste the following code snippet:

```
gameGrid.clear();
locations.clear();
for(int i=0; i<4; i++){
    for(int j=0; j<4; j++){
        Location location = new Location(i,j);
        locations.add(location);
        gameGrid.put(location, null);
    }
}
```

Call initializeGameGrid before startGame

SOLUTION CODE

- *Class*: GameManager
- *Method*: constructor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L65>)
- Copy and paste the following code snippet:

```
initializeGameGrid();
gameGrid.clear();
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 16. Random tiles, random locations


Shuffle a copy of locations list, and add two random tiles at the first two locations of the shuffled list to gameGrid and call redrawTilesInGameGrid

SOLUTION CODE

- *Class*: GameManager
- *Method*: startGame
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L113-118>)

- Copy and paste the following code snippet: ````java List locCopy=locations.stream().collect(Collectors.toList()); Collections.shuffle(locCopy); tileo.setLocation(locCopy.get(0)); gameGrid.put(tileo.getLocation(), tileo); Tile tile1 = Tile.newRandomTile(); tile1.setLocation(locCopy.get(1)); gameGrid.put(tile1.getLocation(), tile1);`

`redrawTilesInGameGrid();`

 Add to the board all valid tiles in gameGrid

```
#### SOLUTION CODE
* *Class*: `GameManager`
* *Method*: `redrawTilesInGameGrid`
* [preview][16.2]
* Copy and paste the following code snippet:
```java
gameGrid.values().stream().filter(Objects::nonNull).forEach(board::addTile);
```

Back to Index (<https://github.com/jperedadnr/Game2048HOL#index>)


---

## STEP 17. How far can a tile go?

Search for the farthest location a tile can be moved in the specified direction, over empty cells and inside the grid

### SOLUTION CODE

- *Class*: `GameManager`
- *Method*: `findFarthestLocation`
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L357-360>)
- Copy and paste the following code snippet:

```
 do {
 farthest = location;
 location = farthest.offset(direction);
} while (location.isValidFor() && gameGrid.get(location)==null);
```

Back to Index (<https://github.com/jperedadnr/Game2048HOL#index>)

---

## STEP 18. A new approach to move the tiles

Instead of using `board.gridGroup` from Step 13 (<https://github.com/jperedadnr/Game2048HOL#step-13->

moving-tiles-on-the-board) to move the tiles, we'll use now `gameGrid`, using a double `IntStream` to traverse the grid.

For every valid tile:

- Find the farthest location possible.
- If it's different from the actual one:
  - Set its layout with `board.moveTile()`
  - Update `gameGrid` in the old location with null
  - Update `gameGrid` with the tile in the new location,
  - Finally set the location of the tile.

*Note: There's an issue with this approach, as the followed order is from top to bottom or left to right. The first tiles can't be moved as the next ones hasn't been moved yet, and they get stuck. This will be addressed later on.*

## SOLUTION CODE

- *Class*: `GameManager`
- *Method*: `move`
- `preview`  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L176-194>)
- Copy and paste the following code snippet:

```
IntStream.range(0, 4).boxed().forEach(i->{
 IntStream.range(0, 4).boxed().forEach(j->{
 Tile t=gameGrid.get(new Location(i,j));
 if(t!=null){
 final Location newLoc=findFarthestLocation(t.getLocation(),direct
ion);
 if(!newLoc.equals(t.getLocation())){
 board.moveTile(t, newLoc);
 gameGrid.put(newLoc, t);
 gameGrid.replace(t.getLocation(),null);
 t.setLocation(newLoc);
 }
 }
 });
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

---

## STEP 19. Animating one tile movement

Animate the tile translation from its actual location to the new one, in 65 ms.

## SOLUTION CODE

- *Class*: GameManager
- *Method*: animateExistingTile
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L375-384>)
- Copy and paste the following code snippet:

```
KeyValue kvX = new KeyValue(tile.layoutXProperty(),
 newLocation.getLayoutX(Board.CELL_SIZE) - (tile.ge
tMinHeight() / 2),
 Interpolator.EASE_OUT);
KeyValue kvY = new KeyValue(tile.layoutYProperty(),
 newLocation.getLayoutY(Board.CELL_SIZE) - (tile.ge
tMinHeight() / 2),
 Interpolator.EASE_OUT);
KeyFrame kfX = new KeyFrame(Duration.millis(65), kvX);
KeyFrame kfY = new KeyFrame(Duration.millis(65), kvY);
timeline.getKeyFrames().add(kfX);
timeline.getKeyFrames().add(kfY);
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

---

## STEP 20. Animating all the tiles together

All the tile animations will be executed at the same time using a `ParallelTransition`. While playing no movement will be allowed. Use the volatile `movingTiles` to exit move when it's true

### SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L147-151>)
- Copy and paste the following code snippet:

```
synchronized (gameGrid) {
 if (movingTiles) {
 return;
 }
}
```

While traversing the grid with valid tiles, add those that are moving to the `parallelTranstion`, instead of using `board.moveTile`

### SOLUTION CODE

- **Class:** GameManager
- **Method:** move
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L186>)
- Copy and paste the following code snippet:  

```
parallelTransition.getChildren().add(animateExistingTile(t, newLoc));
```

Add a listener to find when the animations have finished, and there set `movingTiles` to false

## SOLUTION CODE

- **Class:** GameManager
- **Method:** move
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L296-299>)
- Copy and paste the following code snippet:  

```
parallelTransition.setOnFinished(e -> {
 synchronized (gameGrid) {
 movingTiles = false;
 }
});
```

Set `movingTiles` to true. start the animation and clear de list of animations.

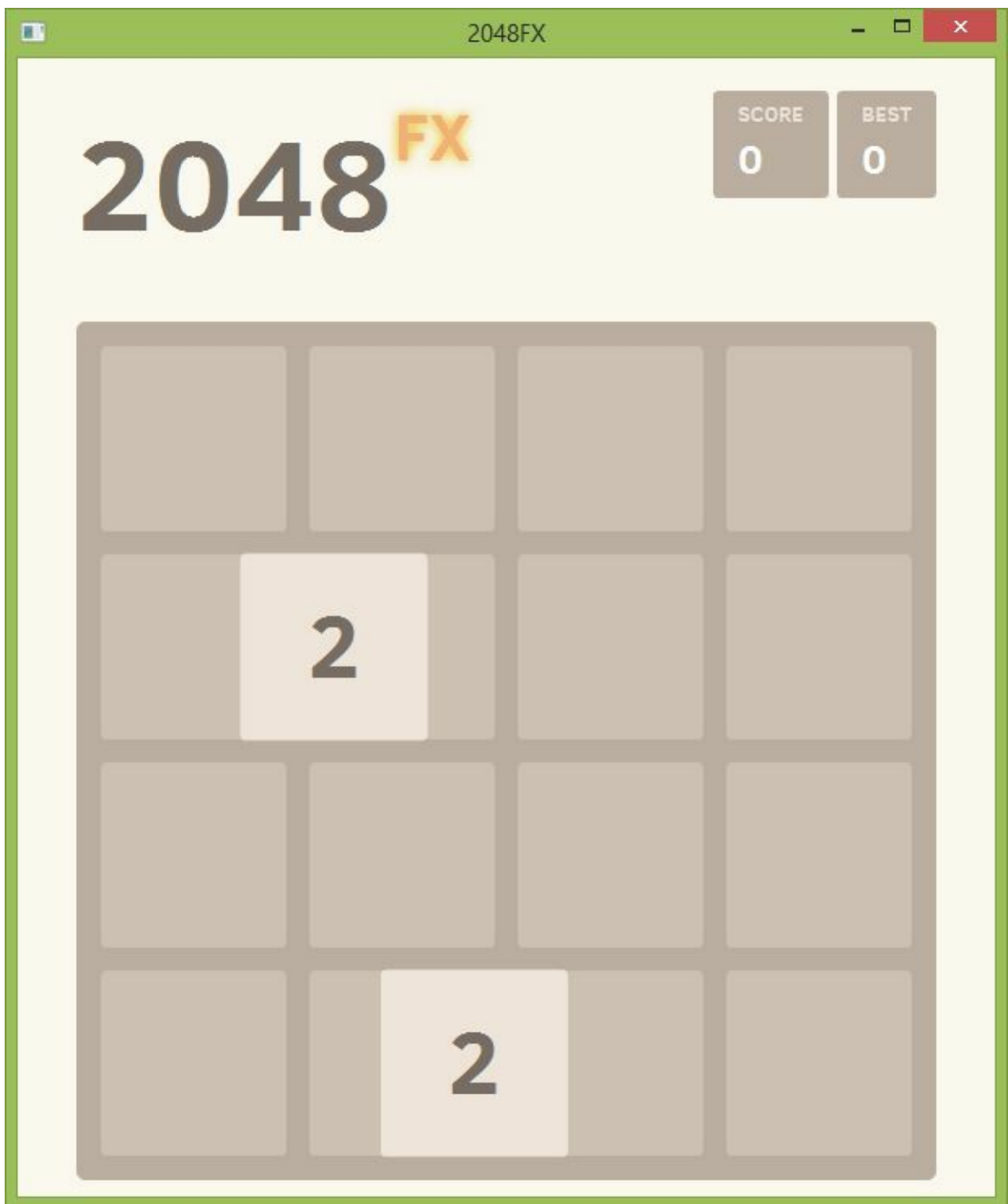
## SOLUTION CODE

- **Class:** GameManager
- **Method:** move
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L338-342>)
- Copy and paste the following code snippet:  

```
synchronized (gameGrid) {
 movingTiles = true;
}
parallelTransition.play();
parallelTransition.getChildren().clear();
```

## Screenshot after #20

Run the project to see the application after completing the first 20 steps. Press an arrow and check the tiles are moving smoothly to the farthest position in the grid



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

---

## STEP 21. Find a random available location


Get a list of available locations on the grid with no tile, shuffle this collection to get a random position, if there's any left, returning the first one in that case

## SOLUTION CODE

- **Class:** GameManager
- **Method:** findRandomAvailableLocation
- preview  
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L399-407>)
- Copy and paste the following code snippet: ````java List availableLocations = locations.stream().filter(l -> gameGrid.get(l) == null) .collect(Collectors.toList());`

`if(availableLocations.isEmpty()) { return null; }`

`Collections.shuffle(availableLocations); location = availableLocations.get(0);`

 Back to [Index][I0]

```

STEP 22. Adding and animating new tiles
Create a new tile at the specified location, with scale set to 0. Add it to the 'board', and to the 'gameGrid'.
Create a 'ScaleTransition' to scale it to 1, in 125 ms, with an easy_out interpolation and play it

SOLUTION CODE
* *Class*: `GameManager`
* *Method*: `addAndAnimateRandomTile`
* [preview][22]
* Copy and paste the following code snippet:
```java
Tile tile = Tile.newRandomTile();
tile.setLocation(randomLocation);
tile.setScaleX(0);
tile.setScaleY(0);
board.addTile(tile);
gameGrid.put(tile.getLocation(), tile);

final ScaleTransition scaleTransition = new ScaleTransition(Duration.millis(125),
    tile);
scaleTransition.setToX(1.0);
scaleTransition.setToY(1.0);
scaleTransition.setInterpolator(Interpolator.EASE_OUT);

scaleTransition.play();
```

Back to Index (<https://github.com/jperedadnr/Game2048HOL#index>)

STEP 23. Playing with new cells on the board

When the parallel transition has finished, get a random location, check if it is not null, and create a random tile, add call `addAndAnimateRandomTile`. Else print "Game Over" for the time being

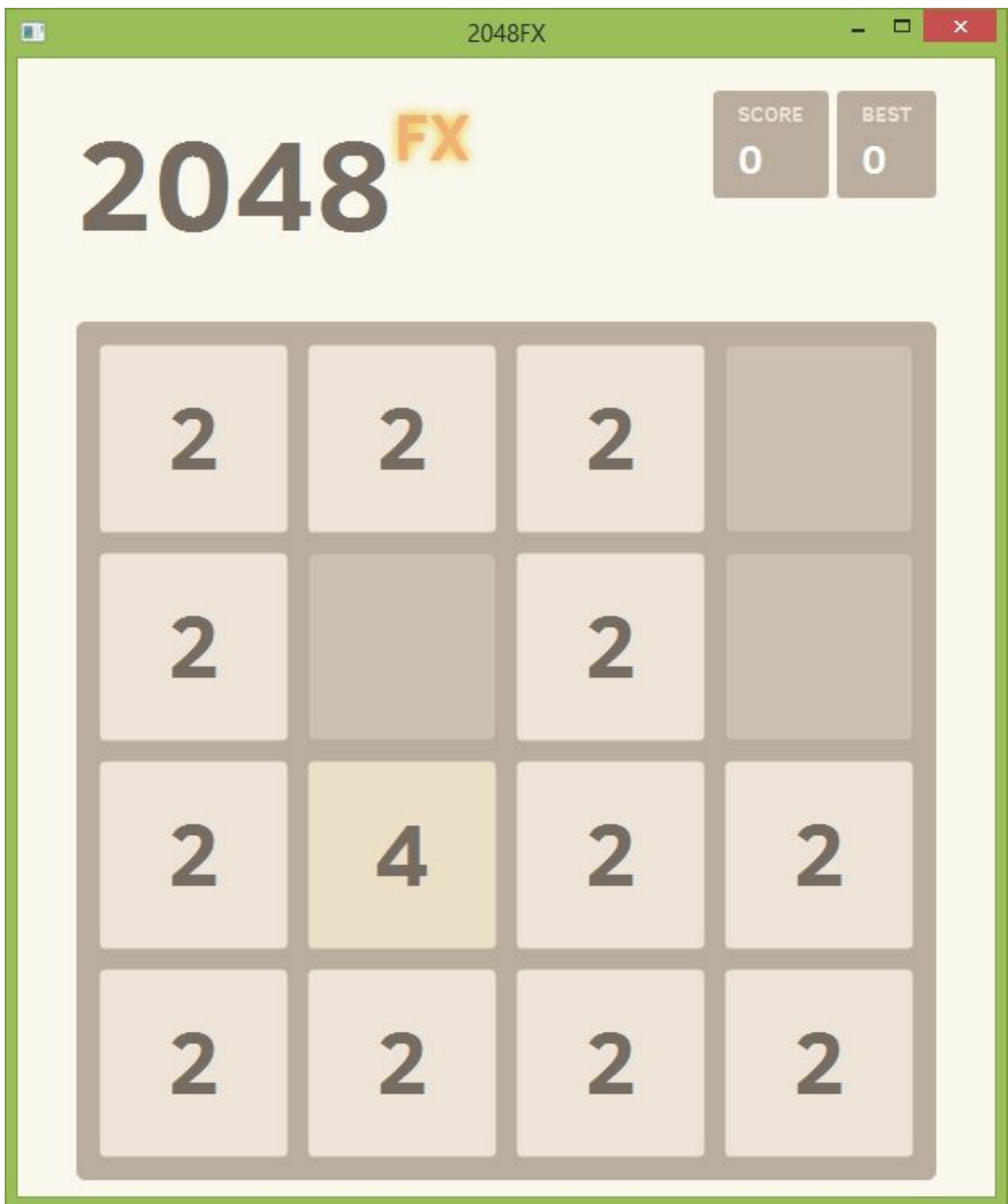
SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L310-323>)
- Copy and paste the following code snippet:

```
Location randomAvailableLocation = findRandomAvailableLocation();
if (randomAvailableLocation != null){
    addAndAnimateRandomTile(randomAvailableLocation);
} else {
    System.out.println("Game Over");
}
```

Screenshot after #23

Run the project to see the application after completing the first 23 steps. Press the arrows in any directions, check the tiles are moving smoothly to the farthest position in the grid, and after each movement a new tile appears



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 24. Introducing the GridOperator

Using the lists `traversalX` and `traversalY`, apply an integer binary operator to every cell on the grid, storing the sum of the results of this functional.

SOLUTION CODE

- *Class*: GridOperator
- *Method*: traverseGrid
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GridOperator.java#L25-29>)
- Copy and paste the following code snippet:

```
traversalX.forEach(x -> {
    traversalY.forEach(y -> {
        at.addAndGet(func.applyAsInt(x, y));
    });
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 25. Traversing the grid properly

Replace the double for loop with the `traverseGrid` method

SOLUTION CODE

- *Class*: GameManager
- *Method*: initializeGameGrid
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L91-96>)
- Copy and paste the following code snippet:

```
GridOperator.traverseGrid((i, j) -> {
    Location location = new Location(i, j);
    locations.add(location);
    gameGrid.put(location, null);
    return 0;
});
```

Replace the `IntStreams` from Step 18 (<https://github.com/jperedadnr/Game2048HOL#step-18-a-new-approach-to-move-the-tiles>) with the `traverseGrid` method. Assign the returned value to `tilesWereMoved` to account for the number of tiles moved

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L208-251>)

- Copy and paste the following code snippet:

```

tilesWereMoved = GridOperator.traverseGrid((i, j) -> {
    Tile t = gameGrid.get(new Location(i, j));
    if (t != null) {
        final Location newLoc = findFarthestLocation(t.getLocation(), direction);
        ;
        if (!newLoc.equals(t.getLocation())) {
            parallelTransition.getChildren().add(animateExistingTile(t, newLoc));
        }
        gameGrid.put(newLoc, t);
        gameGrid.replace(t.getLocation(), null);
        t.setLocation(newLoc);
        return 1;
    }
    return 0;
});

```

When the animations have finished, before adding a tile, check if there were some movements done

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ho1/game2048/GameManager.java#L316-318>)
- Copy and paste the following code snippet:

```

if (tilesWereMoved > 0) {
    addAndAnimateRandomTile(randomAvailableLocation);
}

```

Replace the double for loop with the traverseGrid method to create the cells

SOLUTION CODE

- *Class*: Board
- *Method*: createGrid
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ho1/game2048/Board.java#L193-196>)
- Copy and paste the following code snippet:

```

GridOperator.traverseGrid((i, j) -> {
    gridGroup.getChildren().add(createCell(i, j));
    return 0;
});

```


STEP 26. Sorting lists before moving tiles

Sort the `traversalX` and `traversalY` lists, so for right or down directions these are taken in reverse order. This will solve the problem stated in Step 18

(<https://github.com/jperedadnr/Game2048HOL#step-18-a-new-approach-to-move-the-tiles>)

SOLUTION CODE

- *Class*: `GridOperator`
- *Method*: `sortGrid`
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/GridOperator.java#L38-39>)
- Copy and paste the following code snippet:

```
 Collections.sort(traversalX, direction.equals(Direction.RIGHT) ? Collections.reverseOrder() : Integer::compareTo);  
Collections.sort(traversalY, direction.equals(Direction.DOWN)? Collections.reverseOrder() : Integer::compareTo);
```

Call `sortGrid` before traverse the grid when tiles are moved

SOLUTION CODE

- *Class*: `GameManager`
- *Method*: `move`
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/GameManager.java#L201>)
- Copy and paste the following code snippet:

```
 GridOperator.sortGrid(direction);
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 27. Let's start merging tiles

Add to tile's value the one of the tile to be merged to, set the text of the label with the new value, set the tile as merged and replace the old style `"game-title-"+value` with the new one.

SOLUTION CODE

- *Class*: `Tile`
- *Method*: `merge`

- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/hol/game2048/Tile.java#L83-87>)
- Copy and paste the following code snippet:


```
getStyleClass().remove("game-tile-" + value);
this.value += another.getValue();
setText(Integer.toString(value));
merged = true;
getStyleClass().add("game-tile-" + value);
```

Check if the tile can be merged with another tile by comparing their values

SOLUTION CODE

- *Class*: Tile
- *Method*: isMergeable
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/hol/game2048/Tile.java#L94>)
- Copy and paste the following code snippet:


```
return anotherTile != null && getValue()==anotherTile.getValue();
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 28. Animating the merged tiles

Add a sequential animation, with two scale animations, from 1.0 to 1.2, ease_in interpolation, and from 1.2 to 1.0, ease_out interpolation, in 80 ms each

SOLUTION CODE

- *Class*: GameManager
- *Method*: animateMergedTile
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/hol/game2048/GameManager.java#L457-467>)
- Copy and paste the following code snippet:


```
final ScaleTransition scale0 = new ScaleTransition(Duration.millis(80), tile);
scale0.setToX(1.2);
scale0.setToY(1.2);
scale0.setInterpolator(Interpolator.EASE_IN);
final ScaleTransition scale1 = new ScaleTransition(Duration.millis(80), tile);
scale1.setToX(1.0);
scale1.setToY(1.0);
```

```
scale1.setInterpolator(Interpolator.EASE_OUT);  
return new SequentialTransition(scale0, scale1);
```


[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 29. Performing tiles merging on the move

- Apply an offset in the specified direction to the tile
- Check if this new tile is a valid tile
- Check if the tile is not null, it is not merged and if both tiles can be merged.
- Then:
 - Merge both tiles
 - Move to front the new one
 - Put the new location in gameGrid, with the new tile
 - Replace last location with null in the map
 - Add the old tile to animateExistingTile, and the new one to animateMergedTile.
 - Add old tile to mergedToBeRemoved
 - Return 1

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L215-240>)
- Copy and paste the following code snippet:

```
 Location nextLocation = newLoc.offset(direction);  
Tile tileToBeMerged = nextLocation.isValidFor() ? gameGrid.get(nextLocation)  
: null;  
if (tileToBeMerged != null && !tileToBeMerged.isMerged() && t.isMergeable(  
tileToBeMerged)) {  
    tileToBeMerged.merge(t);  
    tileToBeMerged.toFront();  
    gameGrid.put(nextLocation, tileToBeMerged);  
    gameGrid.replace(t.getLocation(), null);  
    parallelTransition.getChildren().add(animateExistingTile(t, nextLocation))  
;  
    parallelTransition.getChildren().add(animateMergedTile(tileToBeMerged));  
    mergedToBeRemoved.add(t);  
    return 1;  
}
```

STEP 30. Cleaning merged tiles

When the animations have finished, remove the tiles in the set `mergedToBeRemoved` from the `gridGroup` and clear the set. For all the tiles on the board, set to false their merged property

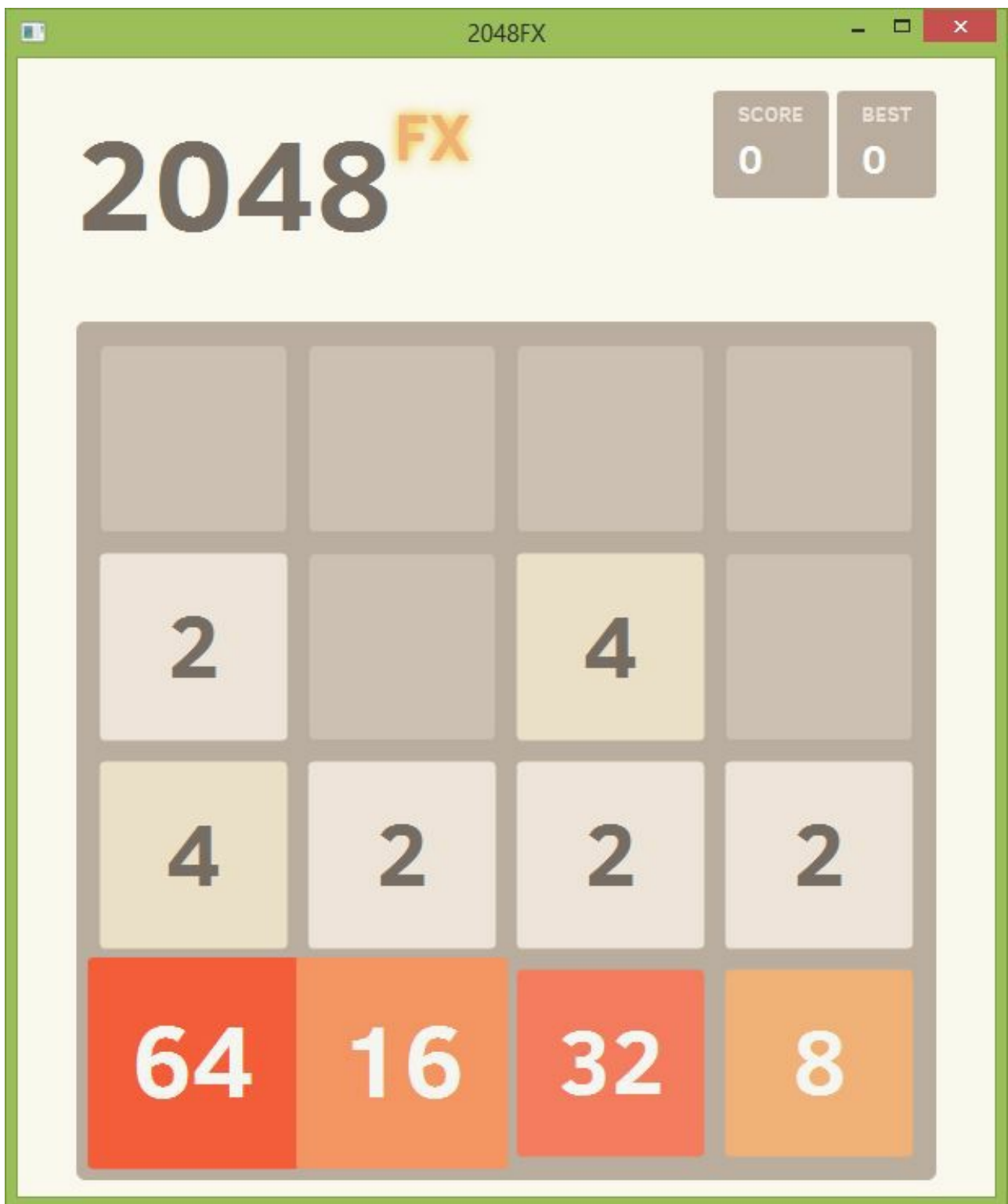
SOLUTION CODE

- *Class*: `GameManager`
- *Method*: `move`
- `preview`
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/GameManager.java#L303-305>)
- Copy and paste the following code snippet:

```
board.getGridGroup().getChildren().removeAll(mergedToBeRemoved);  
mergedToBeRemoved.clear();  
gameGrid.values().stream().filter(Objects::nonNull).forEach(t->t.setMerged(false));
```

Screenshot after #30

Run the project to see the application after completing the first 30 steps. Press the arrows in any directions, check the tiles are moving smoothly to the farthest position in the grid, merging with they neighbours if they can, while after each movement a new tile appears




[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 31. Fixing undesired effects

Dropshadow effects on tiles with values 128+ can displace the grid on the scene. To avoid it, add a rectangle to clip `hBottom`

SOLUTION CODE

- *Class*: Board
- *Method*: createGrid
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L214-215>)
- Copy and paste the following code snippet:

```
 Rectangle rect = new Rectangle(GRID_WIDTH, GRID_WIDTH);
    hBottom.setClip(rect);
```


[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 32. Adding points

Add styles "game-label" and "game-points" for lblPoints. It should have a minimum length of 100, and center alignment. Add it to the board.

SOLUTION CODE

- *Class*: Board
- *Method*: createScore
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L125-128>)
- Copy and paste the following code snippet:

```
 lblPoints.setStyleClass().addAll("game-label", "game-points");
    lblPoints.setAlignment(Pos.CENTER);
    lblPoints.setMinWidth(100);
    getChildren().add(lblPoints);
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 33. Counting points

Bind lblPoints text property to gameMovePoints with a "+" prefix, when points>0. Bind lblScore text property with gameScoreProperty.

SOLUTION CODE

- *Class*: Board
- *Method*: createScore
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L134-137>)

- Copy and paste the following code snippet:

```
lblPoints.textProperty().bind(Bindings.createStringBinding(()->
    (gameMovePoints.get()>0)? "+" .concat(Integer.toString(gameMovePoints.g
et())): "" ,
    gameMovePoints.asObject()));
lblScore.textProperty().bind(gameScoreProperty.asString());
```

Add the points to gameMovePoints and to gameScoreProperty.

SOLUTION CODE

- *Class*: Board
- *Method*: addPoints
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L258-259>)
- Copy and paste the following code snippet:

```
gameMovePoints.set(gameMovePoints.get() + points);
gameScoreProperty.set(gameScoreProperty.get() + points);
```

Reset the points before traversing the grid

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/GameManager.java#L205>)
- Copy and paste the following code snippet:

```
board.setPoints(0);
```

Add the points for every merged cell during any movement

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/GameManager.java#L227>)
- Copy and paste the following code snippet:

```
board.addPoints(tileToBeMerged.getValue());
```

Back to Index (<https://github.com/jperedadnr/Game2048HOL#index>)

STEP 34. Centering the points

Add a listener to changes in `lblPoints` text property, so it gets centered right below the center of `vScore`

SOLUTION CODE

- *Class*: Board
- *Method*: createScore
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Board.java#L142-146>)
- Copy and paste the following code snippet:

```
lblScore.textProperty().addListener((ov, s, s1)->{
    lblPoints.setLayoutX(0);
    double midScoreX=vScore.localToScene(vScore.getWidth()/2d, 0).getX();
    lblPoints.setLayoutX(lblPoints.sceneToLocal(midScoreX, 0).getX()-lblPoints
        .getWidth()/2d);
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 35. Animating the points earned

Create the timeline to translate `lblPoints` in Y from 20 to 100 and reduce its opacity from 1 to 0 in 600 ms.

SOLUTION CODE

- *Class*: Board
- *Method*: createScore
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/Board.java#L151-162>)
- Copy and paste the following code snippet:

```
final KeyValue kv00 = new KeyValue(lblPoints.opacityProperty(), 1);
final KeyValue kvY0 = new KeyValue(lblPoints.layoutYProperty(), 20);
final KeyValue kv01 = new KeyValue(lblPoints.opacityProperty(), 0);
final KeyValue kvY1 = new KeyValue(lblPoints.layoutYProperty(), 100);
final KeyFrame kf00 = new KeyFrame(Duration.ZERO, kv00);
final KeyFrame kfY0 = new KeyFrame(Duration.ZERO, kvY0);
Duration animationDuration = Duration.millis(600);
final KeyFrame kf01 = new KeyFrame(animationDuration, kv01);
final KeyFrame kfY1 = new KeyFrame(animationDuration, kvY1);
animateAddedPoints.getKeyFrames().addAll(kf00, kfY0, kf01, kfY1);
```

Start the animation

SOLUTION CODE

- *Class*: Board
- *Method*: animateScore
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L266>)
- Copy and paste the following code snippet:

```
board.animateAddedPoints().playFromStart();
```

Call board.animateScore after traversing the grid

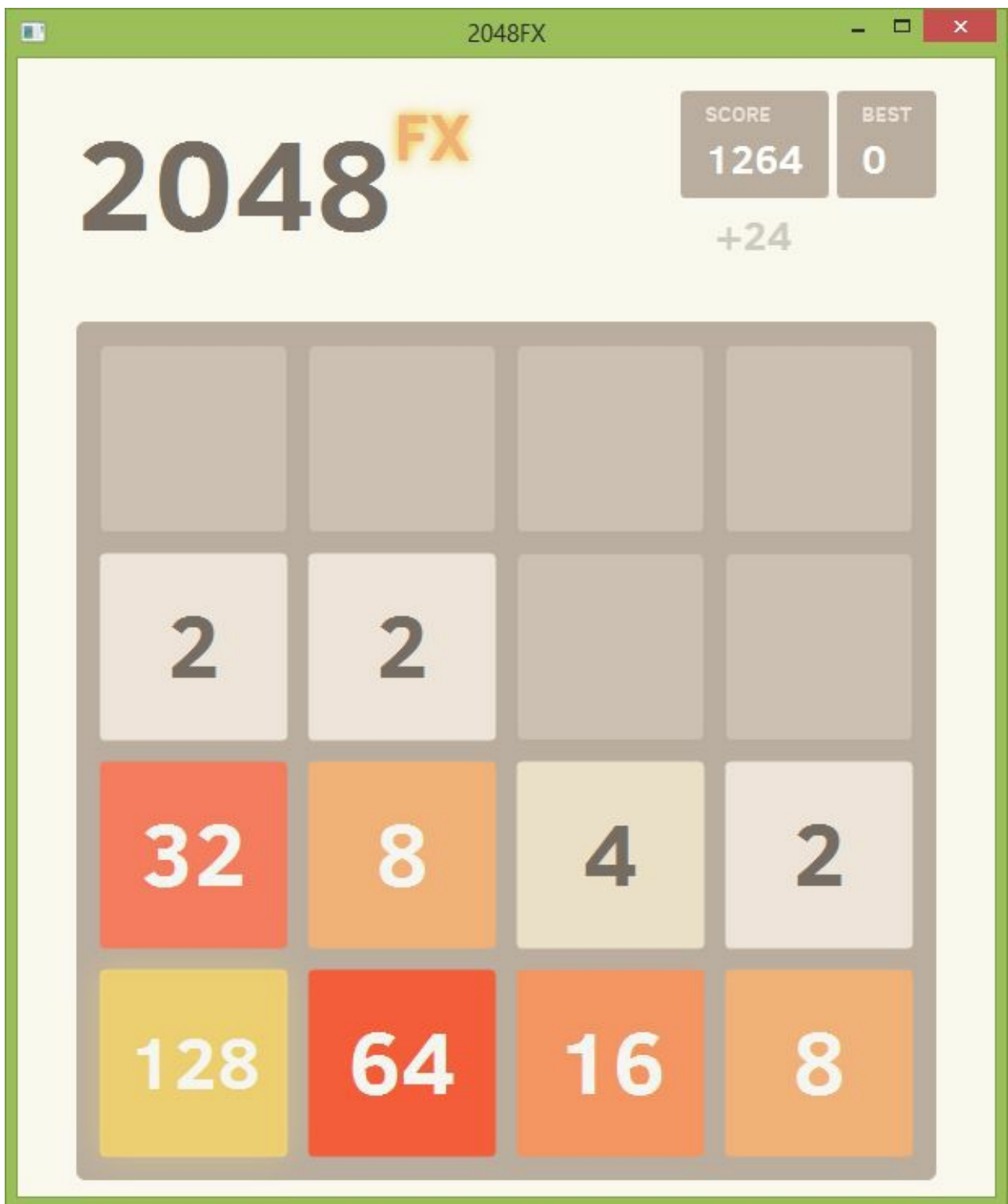
SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/GameManager.java#L291>)
- Copy and paste the following code snippet:

```
board.animateScore();
```

Screenshot after #35

Run the project to see the application after completing the first 35 steps. Press the arrows in any directions, check the tiles are moving smoothly to the farthest position in the grid, merging with their neighbours if they can, while the points of every play are shown below the score, and the total score is increasing



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 36. Looking for pairs of mergeable tiles

Traverse the grid in two directions (up and left, in parallel) and for every tile look if in the next location given by an offset in the specified direction, there is a valid tile mergeable with the former. Increase the account of `numMergeableTile`

SOLUTION CODE

- *Class*: GameManager
- *Method*: mergeMovementsAvailable
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L485-508>)
- Copy and paste the following code snippet:

```
Stream.of(Direction.UP, Direction.LEFT).parallel().forEach(direction -> {
    GridOperator.traverseGrid((x, y) -> {
        Location thisloc = new Location(x, y);
        Tile t1=gameGrid.get(thisloc);
        if(t1!=null){
            Location nextLoc=thisloc.offset(direction);
            if(nextLoc.isValidFor()){
                Tile t2=gameGrid.get(nextLoc);
                if(t2!=null && t1.isMergeable(t2)){
                    numMergeableTile.incrementAndGet();
                }
            }
        }
    })
    return 0;
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 37. Checking for Game Over

When the animation have finished, and randomAvailableLocation is null, call mergeMovementsAvailable and if this returns 0 then print "Game Over"

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L326-331>)
- Copy and paste the following code snippet:

```
if(mergeMovementsAvailable()==0){
    System.out.println("Game Over");
}
```

After the last movement, when the grid is full of tiles, check if there are movements available. In case there are no mergeable tiles, print "Game Over"

SOLUTION CODE

- **Class:** GameManager
- **Method:** addAndAnimateRandomTile
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L433-441>)
- Copy and paste the following code snippet:

```
scaleTransition.setOnFinished(e -> {  
    if (gameGrid.values().parallelStream().noneMatch(Objects::isNull) && mergeMovementsAvailable() == 0 ) {  
        System.out.println("Game Over");  
    }  
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 38. Checking for the winning tile

While traversing the grid, check if the merged tile is 2048, and print "You Win!"

SOLUTION CODE

- **Class:** GameManager
- **Method:** move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L231-237>)
- Copy and paste the following code snippet:

```
if(tileToBeMerged.getValue() == 2048){  
    System.out.println("You win!");  
}
```


[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 39. Setting options buttons

- Style buttons bTry and bContinue with "game-button"
- Add listeners to button onAction property. In both buttons remove overlay and buttonsOverlay. In bTry also remove all tiles in gridGroup. Set resetGame to false and reset all game properties: gameScoreProperty, gameWonProperty and gameOverProperty. Then set resetGame to true

SOLUTION CODE

- *Class*: Board
- *Method*: initGameProperties
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ol/game2048/Board.java#L285-296>)
- Copy and paste the following code snippet:

```
 bTry.getStyleClass().add("game-button");  
bTry.setOnAction(e->{  
    getChildren().removeAll(overlay, buttonsOverlay);  
    gridGroup.getChildren().removeIf(c->c instanceof Tile);  
    resetGame.set(false);  
    gameScoreProperty.set(0);  
    gameWonProperty.set(false);  
    gameOverProperty.set(false);  
    resetGame.set(true);  
});  
bContinue.getStyleClass().add("game-button");  
bContinue.setOnAction(e->getChildren().removeAll(overlay, buttonsOverlay));
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 40. Setting up the Overlay

Add listeners to game over and game won properties.

- For game over:
 - Set style to "game-overlay" and "game-overlay-over"
 - Set `l0vrText` text to "Game Over!"
 - Set `l0vrText` style to "game-label" and "game-lblOver"
 - Add button `bTry`
- For game won:
 - Set style to "game-overlay" and "game-overlay-won"
 - Set `l0vrText` text to "You win!"
 - Set `l0vrText` style to "game-label" and "game-lblWon"
 - Add buttons `bContinue` and `bTry`
- In both cases:
 - Add `overlay` and `buttonsOverlay` to board.

SOLUTION CODE

- *Class*: Board

- **Method:** `initGameProperties`

- preview

(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ho1/game2048/Board.java#L301-319>)

- Copy and paste the following code snippet: ``java
gameOverProperty.addListener((observable, oldValue, newValue) -> { if (newValue) {

```
overlay.getStyleClass().setAll("game-overlay", "game-overlay-over");
lOvrText.setText("Game over!");
lOvrText.getStyleClass().setAll("game-label", "game-lblOver");
buttonsOverlay.getChildren().setAll(bTry);
this.getChildren().addAll(overlay, buttonsOverlay);
} });
```

```
gameWonProperty.addListener((observable, oldValue, newValue) -> { if (newValue) {
overlay.getStyleClass().setAll("game-overlay", "game-overlay-won"); lOvrText.setText("You
win!"); lOvrText.getStyleClass().setAll("game-label", "game-lblWon");
buttonsOverlay.getChildren().setAll(bContinue, bTry);
this.getChildren().addAll(overlay, buttonsOverlay); } });
```

Back to [Index][I0]

```
***
## STEP 41. Listening to game won/over
Call `initGameProperties`

#### SOLUTION CODE
* *Class*: `Board`
* *Method*: constructor
* [preview][41.1]
* Copy and paste the following code snippet:
``java
initGameProperties();
```

Set game won with `board.setGameWin` when the 2048 tile is found

SOLUTION CODE

- **Class:** `GameManager`

- **Method:** `move`

- preview

(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/ho1/game2048/GameManager.java#L235>)

- Copy and paste the following code snippet:

```
board.setGameWin(true);
```

After the animation has finished, set game over with `board.setGameOver` if the grid is full and there are no more mergeable tiles

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L330>)
- Copy and paste the following code snippet:

```
board.setGameOver(true);
```

After the scale animation of a tile, set game over with `board.setGameOver` if the grid is full and there are no more mergeable tiles

SOLUTION CODE

- *Class*: GameManager
- *Method*: addAndAnimateRandomTile
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L438>)
- Copy and paste the following code snippet:

```
board.setGameOver(true);
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 42. Want to try again?

Add a listener to the `resetGameProperty`, so when the button `bTry` is clicked, the game can be initialized again with a clear grid, and started

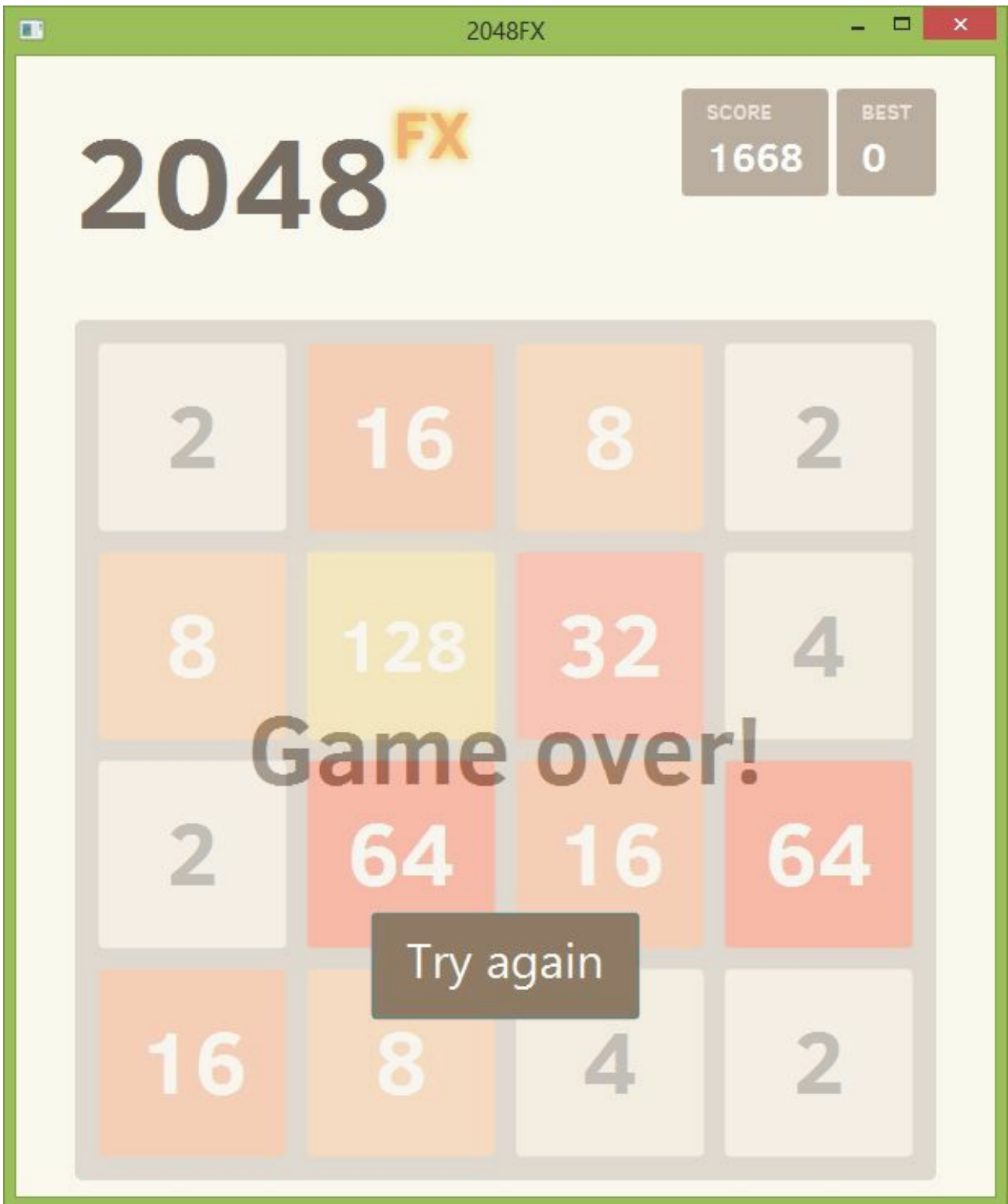
SOLUTION CODE

- *Class*: GameManager
- *Method*: constructor
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L55-60>)
- Copy and paste the following code snippet:

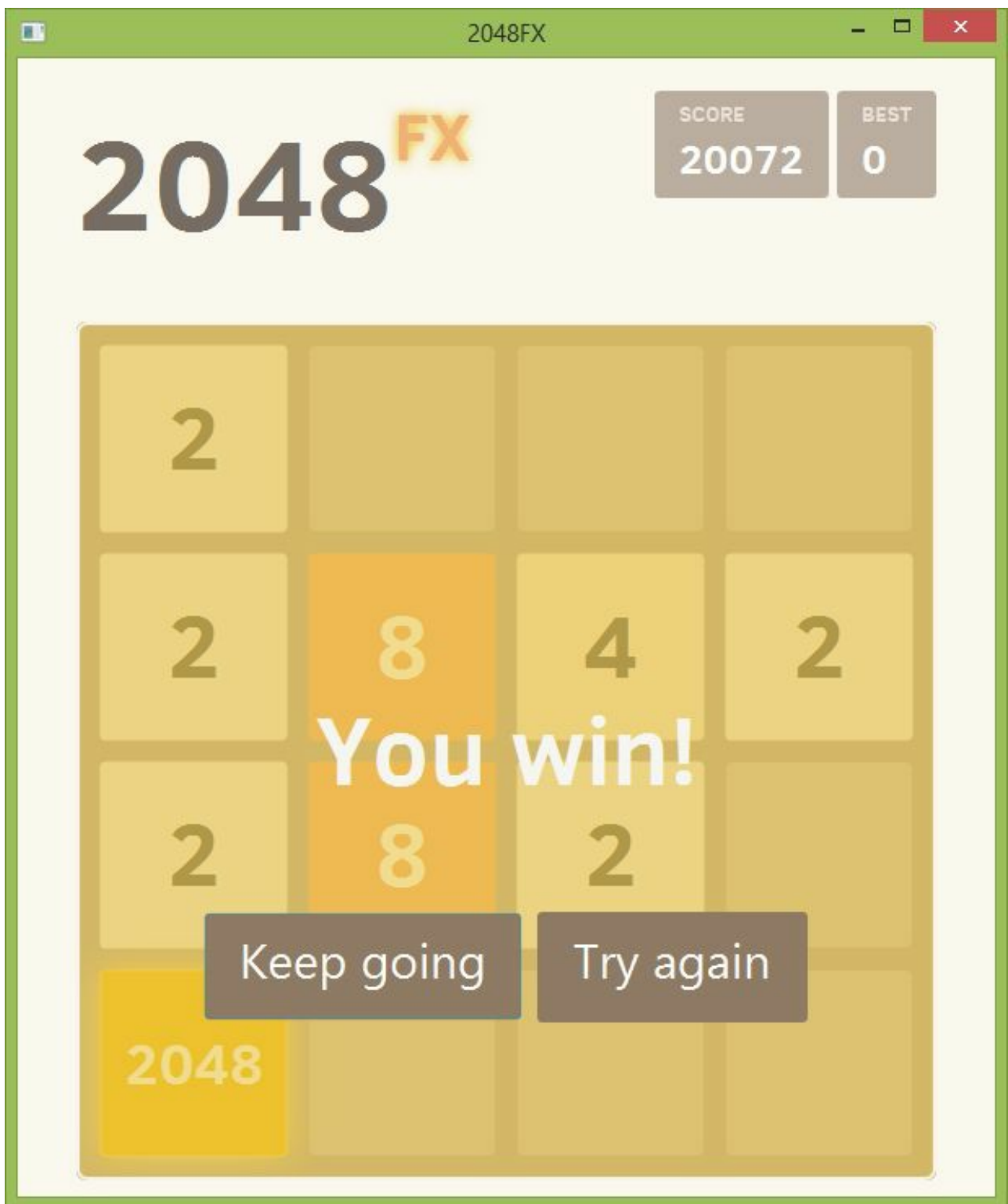
```
board.resetGameProperty().addListener((ov, b, b1) -> {
    if (b1) {
        initializeGameGrid();
        startGame();
    }
});
```

Screenshots after #42

Run the project to see the application after completing the first 42 steps. Move the tiles randomly until you block the board, and the "Game Over!" message shows up. Press "Try again" to start a new game



Try now to win the game



[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 43. Optional tiles

Return an `Optional` of nullable from a given location on the map `gameGrid`

SOLUTION CODE

- *Class*: GameManager
- *Method*: optionalTile
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L523>)
- Copy and paste the following code snippet:

```
> return Optional.ofNullable(gameGrid.get(loc));
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 44. Finding mergeable (optionally null) tiles

Use `optionalTile` to find pairs of mergeable tiles when traversing the grid in two directions, replacing the old code

SOLUTION CODE

- *Class*: GameManager
- *Method*: mergeMovementsAvailable
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L501-504>)
- Copy and paste the following code snippet:

```
> optionalTile(thisloc).ifPresent(t1->{
    optionalTile(thisloc.offset(direction)).filter(t2->t1.isMergeable(t2))
        .ifPresent(t2->numMergeableTile.incrementAndGet());
});
```

[Back to Index \(https://github.com/jperedadnr/Game2048HOL#index\)](https://github.com/jperedadnr/Game2048HOL#index)

STEP 45. Traversing the grid with Optional

Use `optionalTile` to traverse the grid, with an `atomicInteger` to return the results

SOLUTION CODE

- *Class*: GameManager
- *Method*: move
- preview
(<https://github.com/jperedadnr/Game2048Solution/blob/master/src/org/hol/game2048/GameManager.java#L255-285>)
- Copy and paste the following code snippet:

```

tilesWereMoved = GridOperator.traverseGrid((i, j) -> {
    AtomicInteger result=new AtomicInteger();
    optionalTile(new Location(i,j)).ifPresent(t1->{
        final Location newLoc=findFarthestLocation(t1.getLocation(), direction);
        Location nextLocation = newLoc.offset(direction);
        optionalTile(nextLocation).filter(t2->t1.isMergeable(t2) && !t2.isMerged()).ifPresent(t2->{
            t2.merge(t1);
            t2.toFront();
            gameGrid.put(nextLocation, t2);
            gameGrid.replace(t1.getLocation(), null);
            board.addPoints(t2.getValue());
            if(t2.getValue()==2048){
                board.setGameWin(true);
            }
            parallelTransition.getChildren().add(animateExistingTile(t1, nextLocation));
            parallelTransition.getChildren().add(animateMergedTile(t2));
            mergedToBeRemoved.add(t1);

            result.set(1);
        });

        if(result.get()==0 && !newLoc.equals(t1.getLocation())){
            parallelTransition.getChildren().add(animateExistingTile(t1, newLoc));

            gameGrid.put(newLoc, t1);
            gameGrid.replace(t1.getLocation(), null);
            t1.setLocation(newLoc);
            result.set(1);
        }
    });
    return result.get();
});

```

Back to Index (<https://github.com/jperedadnr/Game2048HOL#index>)

STEP 46. Extra features

To find more extra features like:

- Time of play
- Session saving and restoring
- Session pause
- Quit
- Best personal result
- Resizable board

- Any size of cells
- Run on ARM devices
- ...

Go to this URL and fork this repo:

<https://github.com/brunoborges/fx2048> (<https://github.com/brunoborges/fx2048>)

[Back to Index](https://github.com/jperedadnr/Game2048HOL#index) (<https://github.com/jperedadnr/Game2048HOL#index>)
