Maze Game:
Save Christmas

A java swing project realized by

Bálint Roland

2021

Introduction

1. How I got the idea

We were already in December when I realized that I must make this project, and I still have no idea what to make. I wanted something not too simple, yet not too complicated, something I can learn from, and have fun while making it. At first, I had no idea what could be done with a java GUI, or java swing in general. In my head, GUI meant something like a login system, or just a database with an interface.

I remember sitting in the classroom, during Database laboratory, doing my Fundamental Algorithms homework. We had to write a depth first search algorithm, but we already had some code written, which nicely displayed a grid with cells. Suddenly I got the idea, the perfect theme for my OOP project: *a chess table*. So I went on the internet, googled “java chess”, and oh my god, I found some tutorials, some code, even written GitHub projects. I was like this is it, this is possible, it matches the requirements, and I will have a real fun time implementing it.

Up until we went on a brake, and I found out that this was given as an example by the teacher of the other group, and somebody is already making it.

It was hard to say goodbye, but I had to. I need a new idea. Games. What can be done in java and is a game? YouTube: “java games”.

For the next few days, I was searching for games which can be written in java, such as: snake, pong, minesweeper, tic-tac-toe. After a while, I was like: why can’t I make my own game? I just need an idea.

My definition of insane

Let’s create a “cheap replica” of the Hardest Game Ever! If you don’t know what it is, good, ignorance is bliss. It was a Flash game, where you were a little red square and you had to go throgh a maze, with enemies, coins and stuff like that. The idea was all over Flash games (rest in peace). And you could go creative with the maps, it never became boring.

2. Getting to work

Now, the idea is set. I want a maze game. I want a square as a player. I want to have 10 levels at least. Each level would have a different setup. I want some walls: if the player hits them, he/she must do the level again, from start. Then I got more and more ideas. At last, I had these elements:

- *Player*
- *Walls*, which could further be classified as
    - static walls: they do not move
    - flickering walls: they would disappear and reappear constantly
- *Enemies*: they move, and if you hit them, you respawn

Implementation Details

1. First steps

There is of course, a lot of ways to do this. I knew that there are many tools which could help me with the project, but I also knew that I am very new to java, and I want to learn the hard way, so I said: *no tools, just java swing*. During my long-lasting research, I found a channel named Bro Code, which came in handy. Two of his videos got my attention: snake and pong game.

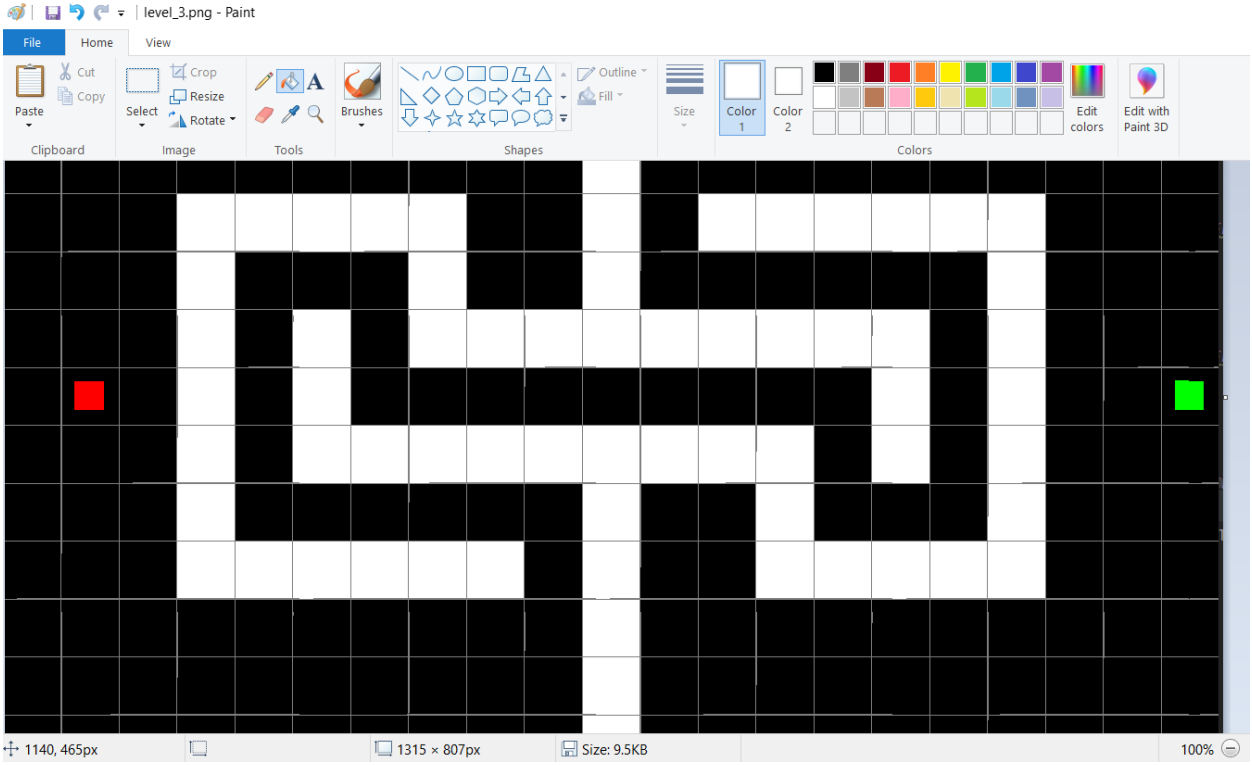
So, since I had never done this before and had no idea at the start how to implement such a thing, I tried to make an analogy from elements of these two games.

My player is like a paddle from the pong game, except it moves in two directions. Enemies are like the player, but they constantly move in a direction, like the snake, and they bounce off things, just like the ball.

There was one big problem: the walls. How could I draw them nicely? Fortunately, I came up with a solution for this as well: while the player can move around freely, walls will have a “limitation”, just like the snake.

So, following my savior Bro Code's steps, I divided the screen on units, and drew the gridlines, so that I could plan the levels easier. His game loop and paint methods worked, and I followed his logic, making each component a different class, adding them to game panel as an instance (or an array of objects), putting them in the draw method, and in the move method if they move.

The first thing I did was designing the levels. Since I had the gridlines, I made a screenshot of the screen, and designed the levels in good old paint.



Evil in progress

I had to draw each wall individually, giving the top left corner's coordinates, the width, and the height. Even though the frame is not resizable, I wanted to make it somewhat adapting, so each wall's coordinates were given with the parameters UNITS_HORIZONTALLY and UNITS_VERTICALLY, and mostly relative to the middle of the screen.

2. Adding some extras

I saw it coming together, and although at this point the game was very plain, it was playable, and it made me very happy. I had to add the other stuff, so I started with the flickering walls. At this time, I had a timer, which was basically another thread, started together with the game loop thread, but it was sleeping for one second continuously, simulating a counter (it is probably not the most elegant solution, but it worked, and it also introduced me to threads). So, in even seconds the walls were drawn with pink and they had collision, in odd seconds they were invisible, aka drawn with black and without collision.

How about enemies? Well, adding them was not too hard, except for their collision: they can collide with the player, with walls, or even with other enemies. Luckily, since they move on only one axis, the bouncing effect can be easily solved.

```
//if they need to bounce, just invert their velocity - one velocity is always 0
if (bounce){
    enemies[i].setXVelocity(-enemies[i].getXVelocity());
    enemies[i].setYVelocity(-enemies[i].getYVelocity());
}
```

Kind of proud of these lines

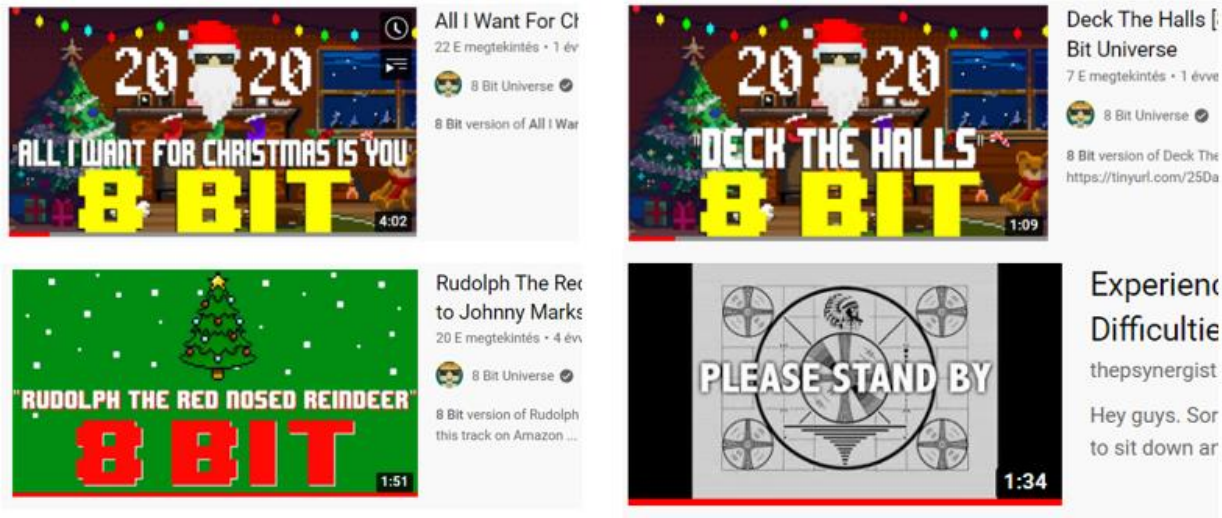
Now I had every element, and I could combine them, but I had to test with different speeds for enemies and for the player. Of course, I wanted to make enemies faster than the player: there is nothing better than seeing someone desperately trying to run away from enemies. Some levels needed adjustment, some even turned out impossible to finish, so I had to add extra walls and remove some enemies.

3. Testing and refining

At this point I would like to thank my sister who was the first tester of this game, and the first person I made mad with this game. Indeed, it can be annoying, especially for people who aren't used to similar video games.

When I showed it to my parents, it had no theme at all. But they were like hey, it is like Santa going for the gift, or something like that (they also said that I should make the player an image of Santa instead of a red square, which is boring, but I couldn't make it- sorry mom and dad).

I was like whatever, let's add some Christmas music in the background then. Of course, it was Mariah Carey's *All I want for Christmas is you*, to make the game even more annoying. And it turned out extra loud, which is a feature, not a bug, I promise. But after a time, I switched it to an 8-bit version of the song and went on a more retro-like Christmas theme.



Music I added to the project - I do not own copyrights, all credits to the amazing artists

4. Adding other panels

At this point, the game itself was ready, but I needed other panels too. After writing together everything needed, I came up with this list of panels:

- Congratulations panel
  - greets the player at the end of the game
  - asks if he/she wants to submit his/her score
  - gives the option to going back to the home screen
- Credits panel
  - a panel for showing off that I made this project
  - also added a link to my GitHub (give me a follow)
- High Scores panel
  - shows previous high scores
  - with arrow buttons the viewer can browse pages of high scores
- Insert High Score panel
  - asks for a name

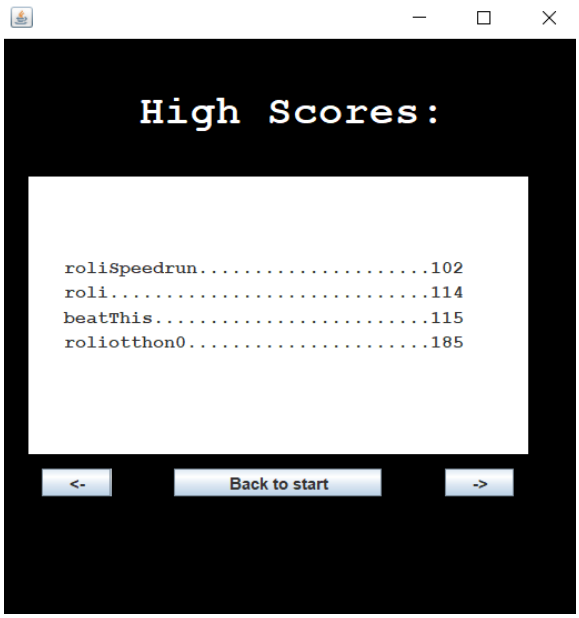
    - it must consist only of digits and letters, and it must be longer than 2 characters
- Story panel
    - a panel showing a short story before starting the came
- Welcome panel
    - first panel showing up when game is started
    - basically, serves as a home page/ home panel

5. Database connection

There is a lot of other things I could talk about, but since my time is also precious, I will move on to some key aspects. I did connect my project to a local database. It has one table, named high_scores.

	id	nickname	total_time	added_on
1	22	roli	114	2021-12-21
2	23	beatThis	115	2021-12-21
3	24	roliotthon0	185	2021-12-22
4	25	roliSpeedrun	102	2021-12-22

Column *id* is the primary key of auto incremented integer, *nickname* is varchar(20), *total_time* is integer (otherwise ordering does not work properly), *added_on* is of type date. With this table, and proper connection set, the High Scores panel looks like the picture on the right.



If the project is run without the database set, it will crash. One way to avoid this is to replace methods showTable and insertRecord of class DatabaseHandler with empty methods. If the space limitations let me, I will also include this version of the game, so it can be played without worrying about the database.

Conclusion

Well, in the end, I can say that it was not easy to get this work, and most probably it is not the nicest code. I may look back on this project years later, saying things like *"Did I really write this?"*. However, at the very moment, I am satisfied with the outcome, and for a first java project, I think it is not bad. After all, I reached my goal: some people got mad playing this. Anyways, my personal record at the time of writing this is 102, try beating it without cheating.

Things I learned: a lot. I met things I never heard of before: threads, regex, card layout, numerous SQL errors, tutorial made by an Indian lady, angry Stack Overflow admins and even Chinese codes from the dark web. It was fun, but I am glad it is done, and I can move on. It is a fun little project, but probably there are better solutions for things I did, more elegant, more practical, and requiring less effort.

Sources, inspirations, links that helped a lot

Bro Code related:

https://www.youtube.com/c/BroCodez

https://www.youtube.com/watch?v=bI6e6qjJ8JQ

https://www.youtube.com/watch?v=oLirZqJFKPE

Database related:

https://www.youtube.com/watch?v=e8g9eNnFpHQ&t=249s

https://www.youtube.com/watch?v=j3XDlNezQ-k&t=501s

Other tutorials on point:

https://www.youtube.com/watch?v=sAReaaTxNGU&t=412s

https://www.youtube.com/watch?v=TErboGLHZGA&t=654s

And of course, the basics:

https://stackoverflow.com/

https://www.geeksforgeeks.org/

Music:

https://www.youtube.com/watch?v=W9OawAwafG0

https://www.youtube.com/watch?v=tR0OLPeDnpI

https://www.youtube.com/watch?v=gJOb06FSFGE

https://www.youtube.com/watch?v=-WWVSC53YCI&t=8s

Pictures:

http://pixelartmaker.com/art/04d1f5ce9f9b009

https://www.vectorstock.com/royalty-free-vector/santa-claus-pixel-art-cartoon-retro-game-style-vector-19117025