Lara Stölting, Jasmin Bödiker und Jennifer Zerbian

Escaperoom Adventure - Where is my Emma?

This project was created as part of the 'Basic Programming in Python' course at the University Osnabrück.

It is based on the concept of an escape room, where you can examine the individual scenes by clicking on the screen. Each scene consists of 1-2 puzzles, only when you solve the respective puzzles you can switch to the next scene.

If you manage to solve all the puzzles, you finish the game and find out where Emma is!

REQUIREMENTS

You can run

```
pip install -r requirements.txt
```

Or install seperately:

Please install pygame.

`pip install pygame` or `conda install pygame`

Please install numpy.

`pip install numpy` or `conda install numpy`

Please install mathplotlib

`pip install mathplotlib` or `conda install mathplotlib`

Please install pandas

`pip install pandas` or `conda install pandas`

Please install scipy.

`pip install scipy` or `conda install scipy`

To start the game excecute the 'main.py' file

```
python main.py
```

HOW TO PLAY

In order to solve our puzzles you need to click on suspicious things on the screen. Sometimes you find a key, sometimes you find or reveal a part of the puzzle that is needed to solve it.

We hid some hints for you, so keep your eyes open, anything can be useful, maybe it helps if you use pen and paper.

If you have the correct code to solve the puzzle you have to either use your keyboard or you have to click on numbers at your screen.

PROGRAMMING JOURNEY

What additional things (libraries, paradigms, etc.) did you learn during the project?

The main new aspect we learned is the `pygame` library. We learned how to display a frame and 'blit' images on it. We learned how to handle user inputs or so-called '`pygame.events`' (keyboard inputs as well as mouseclicks). Further because of the high interactivity we learned the use of threads in python. The concept of threads was already known ('Einführung in die Softwareentwicklung'), so we decided to use them for handling simultaneous processes. With a (for us) new app 'Poti poti' we learned to design pixel-images and display them. The pixel-look helped a lot in identifying the placement of the many 'buttons'. With paint, we could reduce all images to our wanted resolution in pixel*pixel. For design aspects, we also learned to use and load a .tff font and display it on our game. We learned how to handle blocking user input (with blocking while-loop) graphically. We used many global variables as status variables to keep track of players' actions. Here we learned that you need to state at the beginning 'global x' if you want to change x in this function. Access is possible by only referring to x. We learned how to implement background music and play sounds in our game with '`pygame.mixer`'. For the more scientific part of the program I learned how to connect matplotlib with `pygame` and display plots there, use scipy for linear regression and intensified the work with pandas dataframes.

What challenges did you face?

One challenge we faced, was an updating problem. We learned that if file '`a.py`' has a global variable that has been changed, file '`b.py`' , that has an import statement at the beginning: '`from a import *`' the variable is not automatically updated in file `b.py` ' after the change. We approached this problem by printing this variable in both files and recognizing asynchronicity. First, we tried to import it again (after the change was made) and it worked. Later we solved that problem more elegantly by implementing a getter-function (modularity and visibility reasons). Another challenge with the import was the meaningful connection between several .py files. This was clear as we learned that one cannot import file `b.py` in `a.py` and vice versa simultaneously (which is logical because then you could write everything to one file). We approached this by splitting the program concepts into files `startscreen.py, endscreen.py` and `door1.py, door2.py` and `door3.py` at the beginning. Because the starscreen and endscreen should be equal no matter your doorchoice. After implementing and some redundancy we implemented `display_components` which holds functions and status variables that are used by every file (or the majority of files). The 'bigger' function `handle_userinput` was put in one extra file. The output on the resultscreen was put in `resultscreen.py`. A further but quickly solved challenge was the simultaneous gameplay and time display. Because (as already stated in the beginning) the concept was known, we decided to use threads. One (probably the main one, like in java) handles the gameplay and user input and the other one the displaying of time. The First (naive) approach was that everything is handled by the main process but even after setting the fps to 60, one second was incremented every three (or so) because that thread had so many others to handle in between. Another tricky problem was the evaluation of one click as 'one click. We observed that if we click on exactly one pixel and hold it, the click is only counted as one. But if you click and slightly move your mouse the click often is

evaluated as more. Because we first checked if the event '`pygame.mouse.get_pressed()`' happened and if it was a left-click. We assume that this event does not check if the mouse button was released, only that it was or is pressed. We tried solving this by blocking functions like blocking while-loop or time-sleep. These functions prevent the evaluation of further clicks but make the game unnecessary long. After some research, we found another `pygame.event` which made our lives better: `pygame.event.MOUSEBUTTONUP`. When we checked for that, we know that a previous click happened. So now the player can interact with left and right-click, but that is not a problem for our game.

What further additions could be made?

I think the first obvious addition could be increasing the complexity of the rooms task (like in real escape room games) or adding levels/new rooms. If that is not wanted, one could also add a look-around function which lets you look around in the room from different perspectives (also like in real escape room games). We already implemented some zooming functions so equally we could let the user 'turn around' to look at other things in the rooms. We also have implemented one task, where the key to the next door is collected but neither acknowledged nor displayed, so here could an item box be added that displays it. Furthermore, we had the idea of an Avatar which is seen in the loading screen or the `final_words` room, but we did not entirely embed it. Here is room for improvement. In-game we could add more sound effects/interactivity possibilities. But since there is no button object in this library and we were forced to laboriously implement and calculate all button coordinates by ourselves, we would use other libraries/languages for this kind of game development in future projects.

REUSABLE FUNCTIONS

`button`

The `button` function is universal and simulates a button. It checks if the mouse click is in the given coordinates (x, y of upper right button corner) and the hight/width of the button. To make sure we distinguish between buttons with
the same coordinates but different frames it checks with the argument msg.
If all applies the respactive function is called.

`userinput`

The userinput function takes an arbitrary pygame rectangle and uses it as a box/spechebubble to dispay text on. You can limited the the amount of chars and make sure only integers are accepted.

GAME STRUCTURE AND SOLUTIONS (SPOILER)

BEFORE READING FURTHER... PLEASE PLAY THE GAME

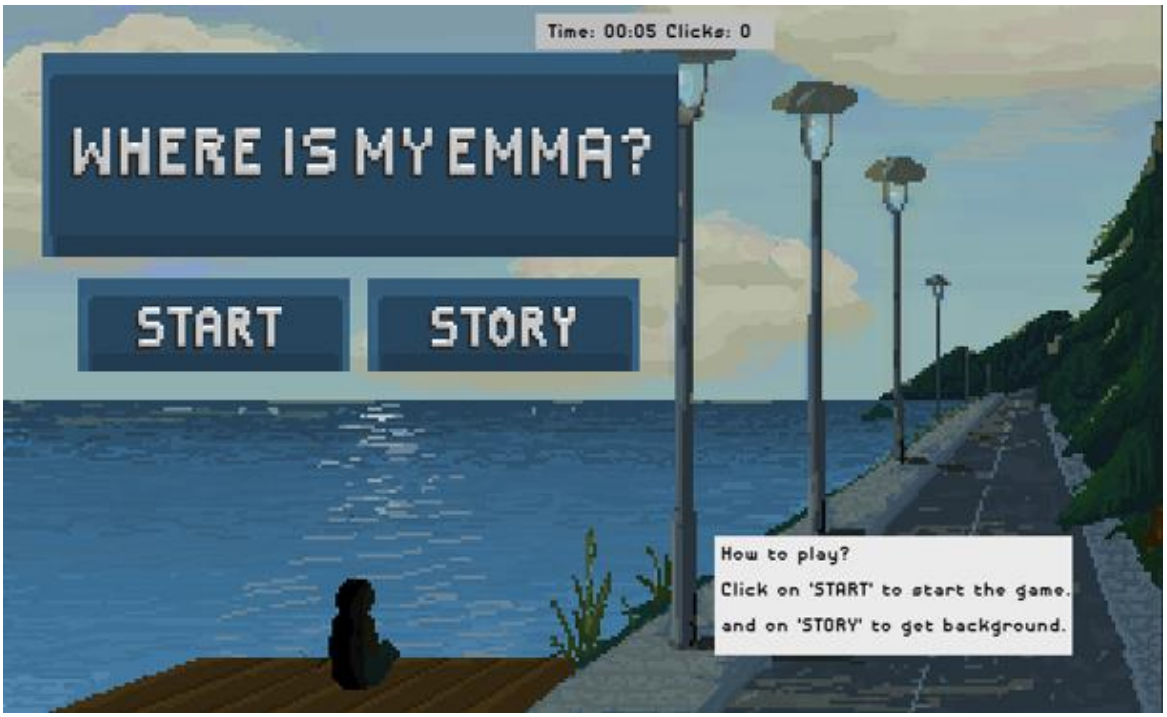
FIRST, IN THIS SECTION WE WILL GIVE YOU THE

SOLUTIONS!

Door 1	Door 2	Door 3
Bathroom	Childsroom	Surprise
Backroom	Garden	
Saferoom	Saferoom	
End	End	

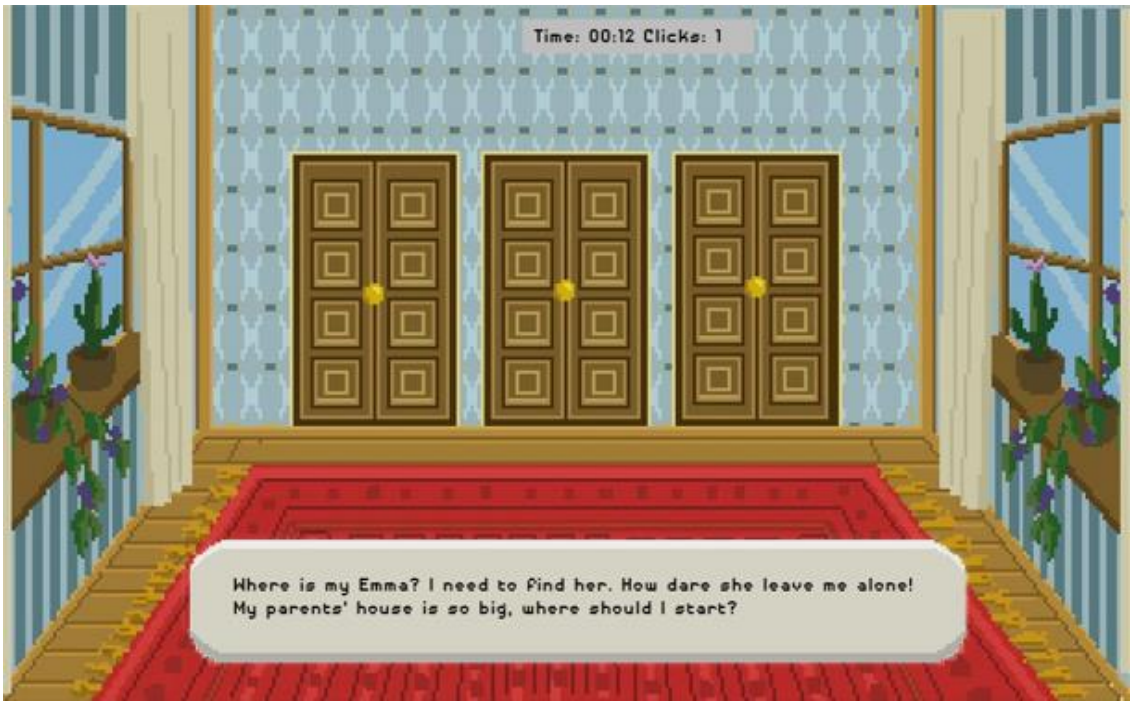
THE STARTING SCREEN

Here you can choose between starting the game or reading the story.



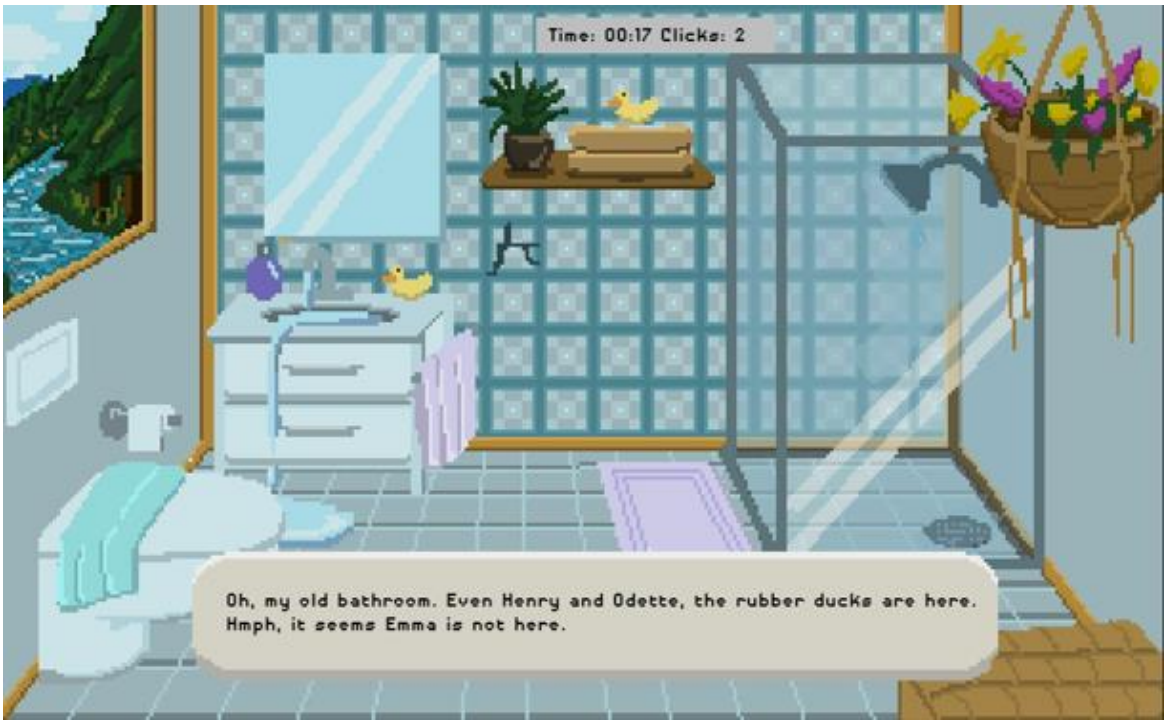
THE BIG DESCISION

If you clicked on start, you will end up in this room where you have to decide between 3 doors.Depending on your choice you will visit different rooms and solve different puzzles.



ROOMS BEHIND DOOR 1

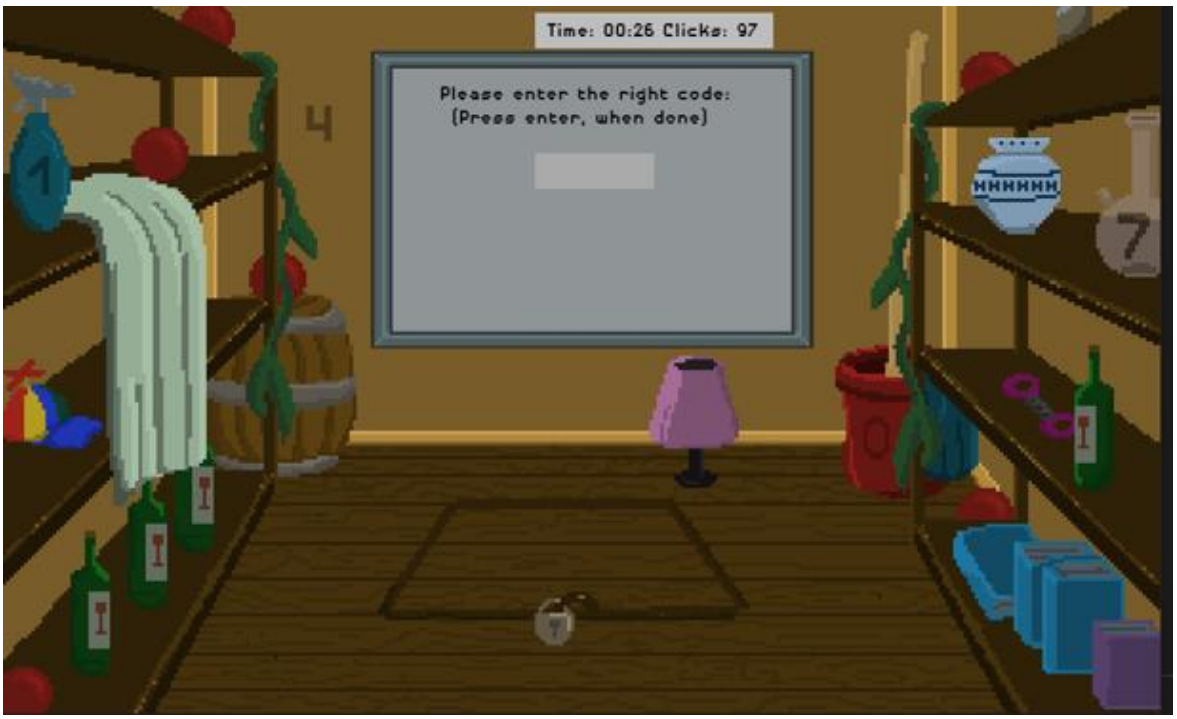
The first room will be the bathroom, try to find out what the puzzle could be.



Crack the tiles on the wall in the middle of the screen by clicking on it.

The second room will be the backroom, here you have to find the correct code, if you want to enter the next room.

Be careful, even the most obvious things can sometimes be wrong!



You can find the key in the vase on the right-hand side.

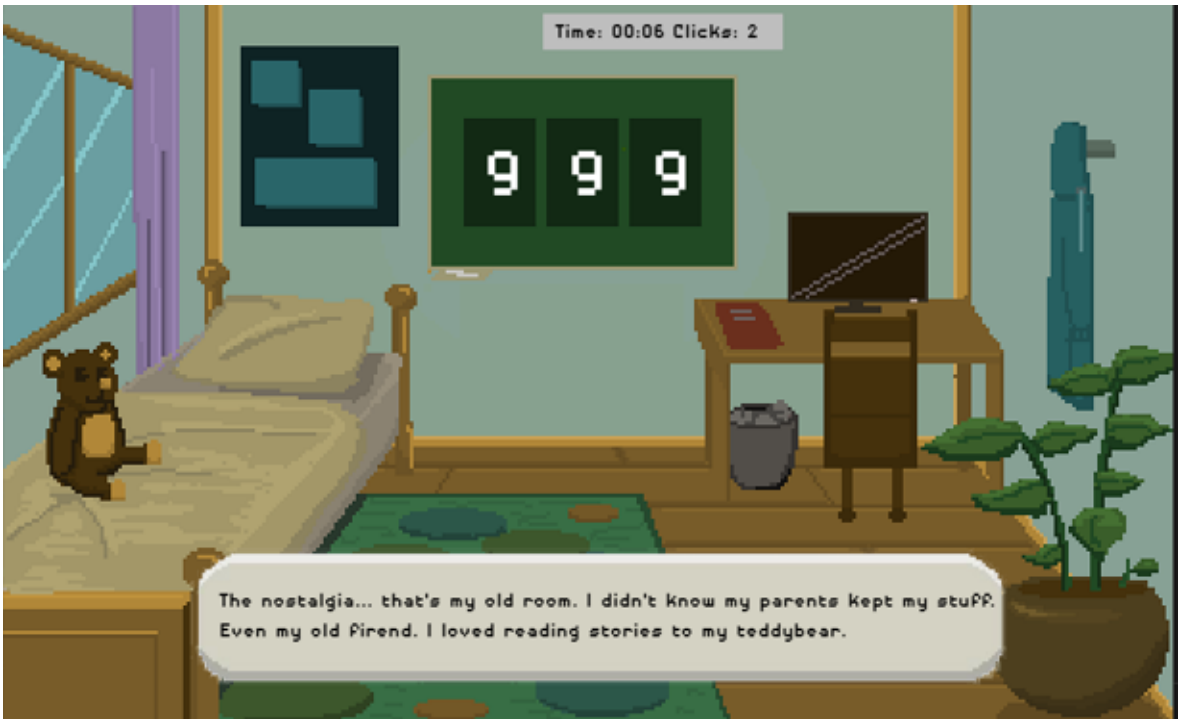
The correct amount of red balls is 15. Please use your keyboard to enter this number (and press enter).

The last room contains a safe. The code can be found in room 2.

You can find the code for the safe in the backroom. It is 1407. This time you have to click on the right numbers on the touchpad.

ROOMS BEHIND DOOR 2

The first room will be the childsroom, here you have to solve a puzzle that is a little bit harder, it consists of a logic puzzle that needs to be solved in order to enter the next room.

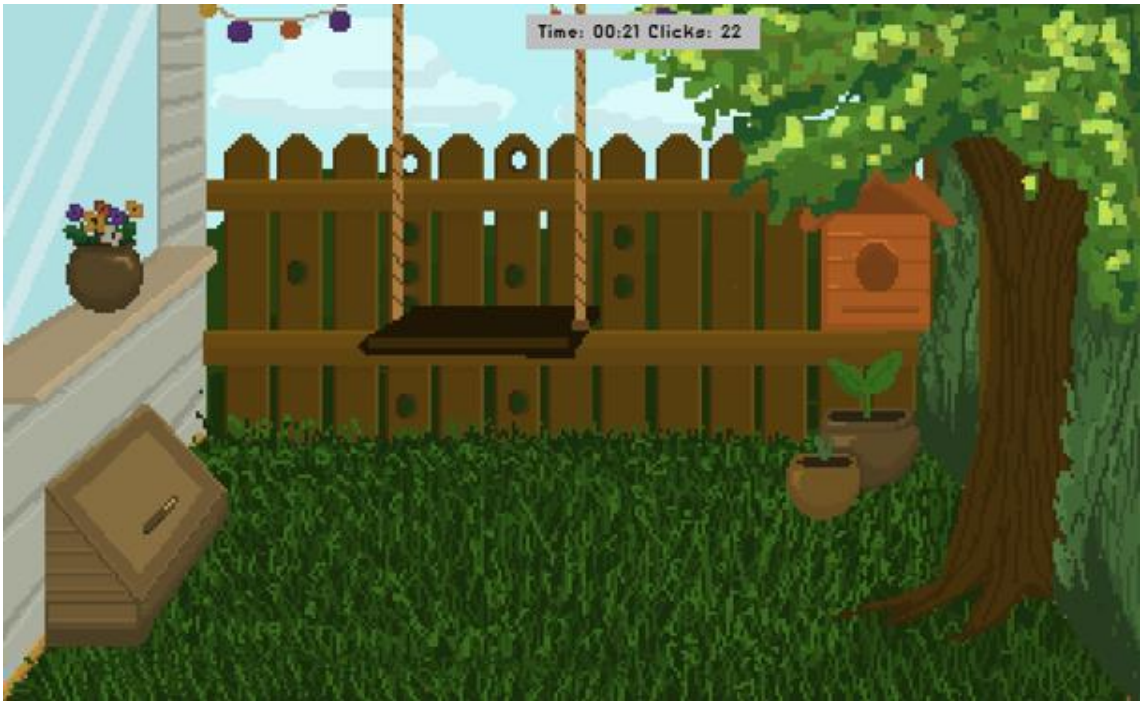


The puzzle can be found in the red book. The solution is 420.

Please click on the numbers on the wall to enter the correct sequence.

The next room is the garden, here you can find a key that is needed if you want to enter the last room.

Please keep in mind that the code for the last room is hidden in this room, and that you cannot come back.



The key can be found in the birdhouse.

The last room consists of a safe. The code can be found in room 2.

You can find the code for the safe on the fence in the garden. It is 1532 (amount of holes).

ROOMS BEHIND DOOR 3

Surprise!

THE STORY EXPLAINED

Our protagonist is a drug addict that comes from his parents funeral when he experiences withdrawal symptoms. Halucinating, he is set on finding Emma. But who or what is Emma? His Sister or something else entirely?

In his search he is headed to his childhood home he hasn´t visited for years.

In the ending of the first room the safe contains a tombstone with the name Emma on it. Realizing it was a haluzination and that he imagined having a sister that died he calls his dealer Escopub to get a new dose to forget everything.

In the ending of the secound room the safe contains a bag of ecstasy/Emma pills. Seeing them he realizes he does not want to go through the cicle of withdrawals and hights again. He gets rid of the bag and calls Escopub for a final goodbye.

SOURCES

- Soundeffects: https://heidruns-musikerseiten.de/downloads/wav-download
- https://pixabay.com/de/sound-effects/
- Music : https://www.youtube.com/watch?v=nod3Gx3c3Kw&ab_channel=rickleal135
- Puzzle (Childsroom): https://escaperoomspiele.com/das-code-logical/