# Brave Heart game

Olexandr Burakov
ČVUT–FIT
burakole@fit.cvut.cz

5. ledna 2023

## 1 Introduction

This report informs about the semestral work created for the BI-PYT course in the winter semester 2022/2023.
I chose a Game project for myself using the extensive pygame library which, as the name implies, has a wide range of tools for creating different games, both 2D and 3D.
The purpose of this work was to create, as far as possible in the realities of studying at the university, a full-fledged 2D graphic game with a map, the main character, monsters, weapons, spells, and various applied techniques. The main goal of the game is to improve the abilities of the main character, kill monsters, and kill the boss.
I wanted to work with this topic again since my first programming project in my life was also a pygame game – I created a space 2D shooter and at that time I lacked neither knowledge nor experience.

## 2 The main used techniques and algorithms

In this section I will talk about the algorithms and techniques used when creating the game, some of them have already been implemented by the creators of pygame, the rest were developed by myself.

### 2.1 Sprites: player and monsters

For monster sprites and the main character creation I used the pygame.sprite module built into pygame. On the basis of this class a programmer should create own sprite classes and objects already from them. Thus, the pygame.sprite.Sprite class plays the role of a kind of abstract class. Although it is not abstract, it is possible to create objects directly from Sprite.
Its ready-made functionality was mainly used to update the rendering of sprites on the screen and to process collisions with other sprites, which this module also allows us to combine in groups – pygame.sprite.Group().

### 2.2 Map

For the graphic component of the map I used sets of icons and pictures, tilesets, of one digital artist from the Internet, I will indicate his link in the sources of the work. Subsequently, I processed individual images manually in Photoshop and adjusted them to the desired dimensions.
To create a complete map, I used a specialized program Tiled, in which I determined the boundaries of sprite movement and also exposed different barrier sprites such as stumps, trees, monuments, fountains, and stones, which have their own hitbox. Sprites can't move through such objects.
The layout of the map is saved in the format .csv, for reading these files was used the csv.reader module which allows to load a map into the program as an object and then read it line by line based on the values at each position.

### 2.3 Weapons: sword and spells

Both the main character's sword and his spells are also sprites in the game itself. Game track their collisions with other sprites, the time of their use, which is controlled by using the pygame.time module, as well as the damage they cause to both the player with spells and monster attacks, and the player's weapons.
When casting spells, visual effects are played on the screen such as fire with a fireball spell, green aura with a cure spell, and other monster spells.

### 2.4 Game control

Keyboard input is used to control the mechanics of the game. Pygame makes it possible to read keystrokes by tracking events and recognizing their types using the pygame.event.get() function and using pygame.key.getpressed().The first function checks during the game if the user is exiting it, the second function tracks keystrokes to move the character, pressing buttons in the menu and changing the type of weapon.
Keys on the keybord to control the game:
- Start menu: Up, Down.

- Game level: Up, Down, Left, Right, A(attack), S(cast spell), W(switch spell), M(ability menu).
- Ability menu: Left, Right, Space(upgrade).

### 2.5 Main character's abilities

As for me, the most interesting part of the game is the character's ability system. First of all, he has the main indicators of health (the red stripe on top) and energy (the blue stripe on top), as well as experience points for killing monsters (the window on the bottom right). Health is spent when getting damage from monsters, energy is spent when casting spells, and experience is spent when improving abilities in the menu.
When upgrading experience, it can be converted in the menu into an improvement of one or another ability of the main character – the cost of improvement in units of experience is written at the bottom of each ability. You can gradually pump yourself the maximum scale of health, energy, attack power, the power of spells cast, and the speed of the character. The more an ability is improved, the more its next improvement will cost, and the better this ability will manifest itself in the game.

### 2.6 Music and sounds

This mechanics of the game has become a stumbling block for me. Pygame also makes it possible to use the built-in module – pygame.mixer in this case. With its help music plays in the game and in the start menu. In addition, I have implemented the sounds of attacks and spells, I will tell you about them in the next chapter, for what reason they are not heard during the gameplay.

## 3 Results

I managed to implement all the basic mechanics and features of the game, as the development of the game progressed, the vision of some aspects changed, but in the end it turned out to implement all the ideas.
As I said at the end of the last chapter, the unsolved problem for me was the simultaneous playback of several musical tracks in the game. The pygame.mixer module have a restriction for playing only 1 audio track using the usual method, but in addition to the main background music, the sounds of fight and spells should also appear in the game. Various methods and functions of this module have been tried, for example, splitting the sound into several sound channels independent of each other, but with this approach I came across the same limitation.

## 4 Conclusion

It was very interesting for me to work with this wonderful library again and be a game developer for a while, believe me, the process is fascinating. I had a chance to work both with the graphic component and various auxiliary programs, like Photoshop and Tiled, and directly with the writing of mechanics, functions and classes generalizing them. I updated in memory what I already knew with experience, and also learned for myself several new modules for me like csv and pygame.mixer.
In the game in the future is possibility to refine the sound component, work on expanding the map and, for example, a variety of monsters – these are the moments that I would do if I had more time.

## Reference


[1] csv module documentation. Online. `https://realpython.com/python-csv/`.

[2] Examples of modules using. Online. `https://www.geeksforgeeks.org`.

[3] Official pygame documentation. Online. `https://www.pygame.org/docs/`.

[4] Official python documentation. Online. `https://docs.python.org/3.11/`.

[5] Official sys module documentation. Online. `https://docs.python.org/3/library/sys.html`.

[6] Pictures, tilesets, and sounds author. Online. `https://pixel-boy.itch.io/ninja-adventure-asset-pack`.

[7] Tiled grpahical application. Online. `https://www.mapeditor.org`.

[8] Youtube video of using tiled. Online. `https://www.youtube.com/watch?v=IHmF_bRpOAE`.