Universidad Carlos III de Madrid

Herranz Belén, 100495930
Sánchez, Izan, 100495774

Programming videogame report

Abstract

This project focuses on trying to simulate the first level of the famous game “1942”. Following all the advice given by teachers and all the rules dictated, a very well organized approximation of the actual game has been created.

1. Project description

All the information related with the project, such as: general information or detailed descriptions about classes and methods used, can be found in this part.

1.1. Overview

First of all, it is crucial to understand the main purpose in the game, which is: destroying all the enemies the user can and, consequently, gaining the highest number of points possible. Then, to have a great overview, it is important to briefly explain the general structure of the program. The most significant area is the board, where all the elements appear, being those: the player’s plane, all the enemies, bullets and power ups. Apart from that, this design contains a starting menu which has three options: running the game, quitting it or looking at the info related to controls, lives or points; and a game over scene that allows you to start over all the program.

1.2. Detailed classes description

Phyton’s pyxel package has been used in order to represent all the animated objects and the game itself. To organize the code, we have created 10 classes with different type of information in each of them:

a) Board class

The majority of the code has been invoked in this class, in order to relate all classes and show them to the program. In the update you can find all the information related to controls and collisions. In addition, all the code related to scenes has been implemented in the draw function of this class.

b) Plane class

In this class we have stored all the code necessary to make the plane move in different directions, to shoot (invoking the bullets class) and, by creating conditionals, to be drawn.

c) Enemy classes

We have created 5 classes, one for each type of enemy. The code is organized similarly in each of them. First, all the variables necessary to implement the game are initialized in the init function, and then, all the information related with movement and drawings can be found in the update and draw functions. Finally, all these classes collapse in a general one called “Enemies”, where they will be invoked with some specific conditions.

d) Bullet classes

Two simple classes related to this point have been created in order to avoid repetition of code, one of them related with player’s bullets and the other with enemy bullets. Furthermore, each one has different sprite calls and updates since they belong to different parts of the game. For example, Bullets related to the plane have a bonus that consist of the following: If the user kills all the enemies of the second type, a bonus will appear on screen and, if catched, it will give a double shot.

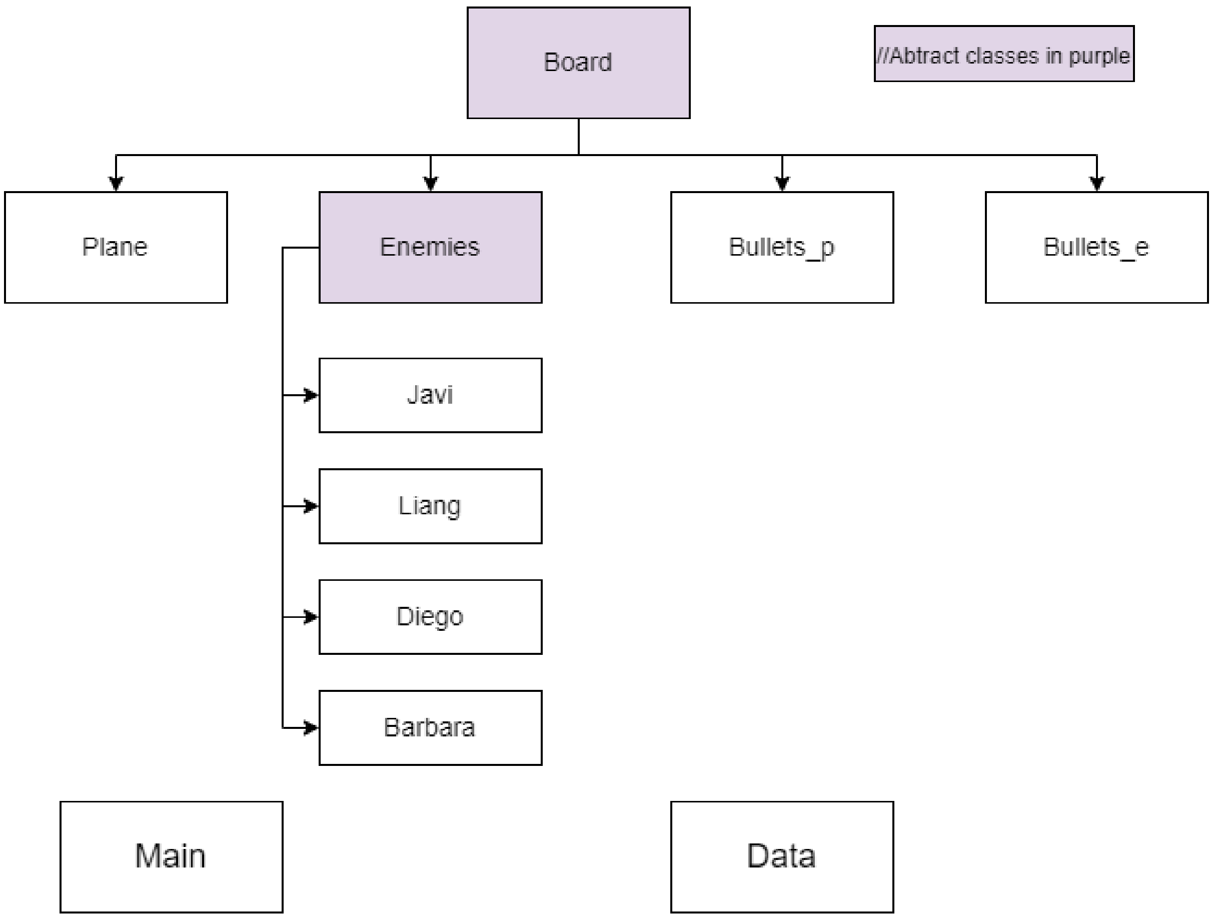
e) Data file

In addition to all classes, we have created a “.py” with all the constants needed in the program, in order to make sudden changes in data a lot easier, and not to commit any mistake while writing numbers in code.

f) Main file

This file is in charge of running the whole code implemented before. To do this, it creates an object, initialized as the class board, that is introduced in the command *“pyxel.run(... , ...)”* in order to run the whole game. This implementation can be done thanks to the organization of all classes, but specifically thanks to invoking them in the main class of the program, the board.

Classes scheme



1.3 Important algorithms

There are some algorithms that are very important in the implementation of the program and need to be defined.

First of all, ***pyxel.btn()*** or ***pyxel.btnp()*** are commands that determine whether the user is pressing a key from the keyboard or not and have been crucial in order to develop the player's plane movement. Other important algorithm is ***pyxel.blt(),*** which is able to take one image from pyxel editor and show it on screen at any coordinates the user wants. This command is responsible for the whole project design.

Apart from commands and algorithms related directly with pyxel, we have used others, such as: ***time.time()*** or ***.append, .remove*** and ***.clear,*** which refer to list methods. The time algorithm has been used in order to create certain animations related to explosions or loopings in an easier way and the lists methods have been very useful, mainly when the program has to reset the whole game and we need to take out all enemies and bullets.

2. Explanation of the game

When running the program a title scene is presented. After that, the user is able to decide between three options: start playing, looking at all the features about the game or not playing. If the user presses the key "F", the program will show another scene with the basic information needed to play the game and, again, the option of starting the game or not.

Then, moving to the main part of the program, if he/she chooses starting it, the game begins. First, the user is only able to see the background moving in an infinite loop and the plane, which is able to move at any moment. Then, after some time enemies start appearing, and we have different types:

The first enemies to appear are the basic ones. They have one life and if you destroy them, shooting bullets by pressing the key space, they will give 10 points. Their movement is very simple, consisting of an infinite diagonal movement.

The second type are the “red enemies”. As the basic ones, they have one life and give 10 points, but their main characteristic is the movement, instead of diagonals, they perform circles.

Then, “bombardiers” will appear. Unlike the other ones, it takes 5 shots to destroy them and they give 20 points per shot. In addition, they can destroy you by simply touching your plane.

Finally, the most difficult enemy of all, the “super-bombardiers”. Instead of having 5 lifes, they have 15 and additionally they shot several bullets at the user. Furthermore its movement is very simple: It goes down vertically until it reaches some height, then it starts going side to side and shooting.

To finish with this part of the report, we can refer to bonuses. In our game the user has one principal bonus, explained before, that allows the plane to shoot 4 bullets instead of only two for 10 seconds. It is obtained every time all red enemies are destroyed. Furthermore, the last important characteristic is another bonus that the user can receive only two times. With this last bonus the plane will perform a loop and become invincible for 1 second. It is obtained by clicking the key “Z”.

3. Conclusion

In order to conclude with this report, we can explain briefly some problems faced during the whole process.

Obviously, the first problem was the lack of information about pyxel, that made us doubt about any command or reference to pyxel editor and slowed the process down.

Another problem was the structure of classes, because it was very difficult to realize whether a class was needed or not, the functionality of the board class or the elements needed in the bullets class.

Finally, the most difficult problem to be solved was the appearance of explosions or the creation of animations, such as loops, mainly because of the relation with the *time.time()* algorithm.

To sum up, this project has been great to explore new aspects of programming in a practical way. Also it has opened our minds to start working as a team, a crucial tool for our career.