# Software Engineering: Project Report

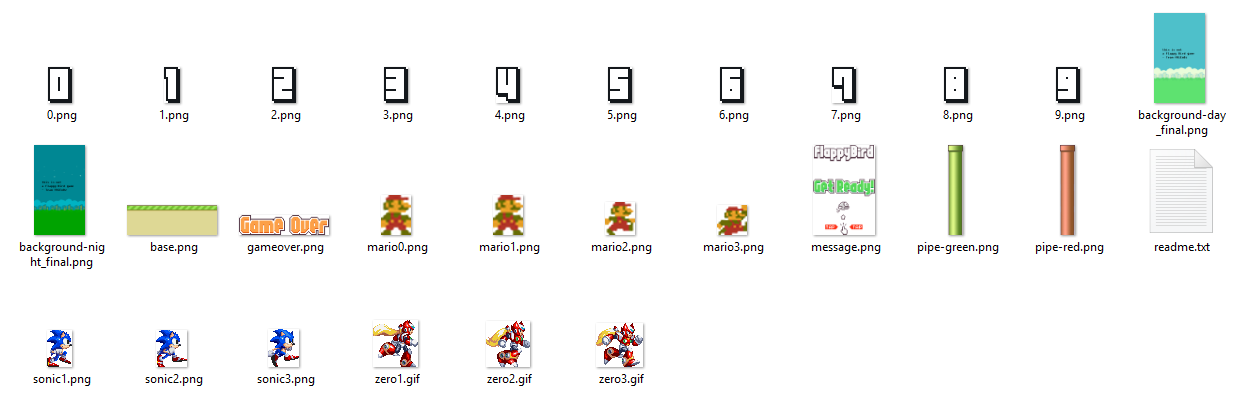
# Team fRiEnDz

Our team has finished the project of Software Engineering. This is the report of the project. The project is to make an improved version of a famous Flappy Bird game. We use Python 3.6.5 with the framework pygame. We will divide this report sections: Process, Specifications, and Testing.

1. Process

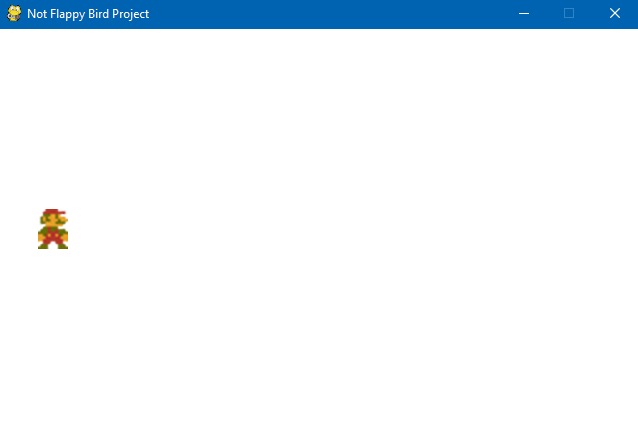
Our team use the traditional process model Waterfall, which is a relatively linear sequential design approach. It tends to be among the less iterative and flexible approaches, as progress flows in largely one direction.

First, we collect all of the requirements for the project. We collect pictures of the game, which is called sprites from the internet. We changed the game a little bit by adding some different characters, such as Zero from Megaman X and Sonic from the game Sonic. Here are all the sprites, backgrounds, pipes… from the folder “assets”:



In the original game, there is just one bird flapping over and over again. We randomly switch between different characters, as we can see here, the three characters are Sonic, Zero and Mario.

As a group, we continuously discuss with each other about what will we do next, then each of us will do a small task of the goal. For example, when we want to load the list of sprites into the game, one guy will find those sprites, another will implement those sprites into the game, here is the basic character with no background and sprites:



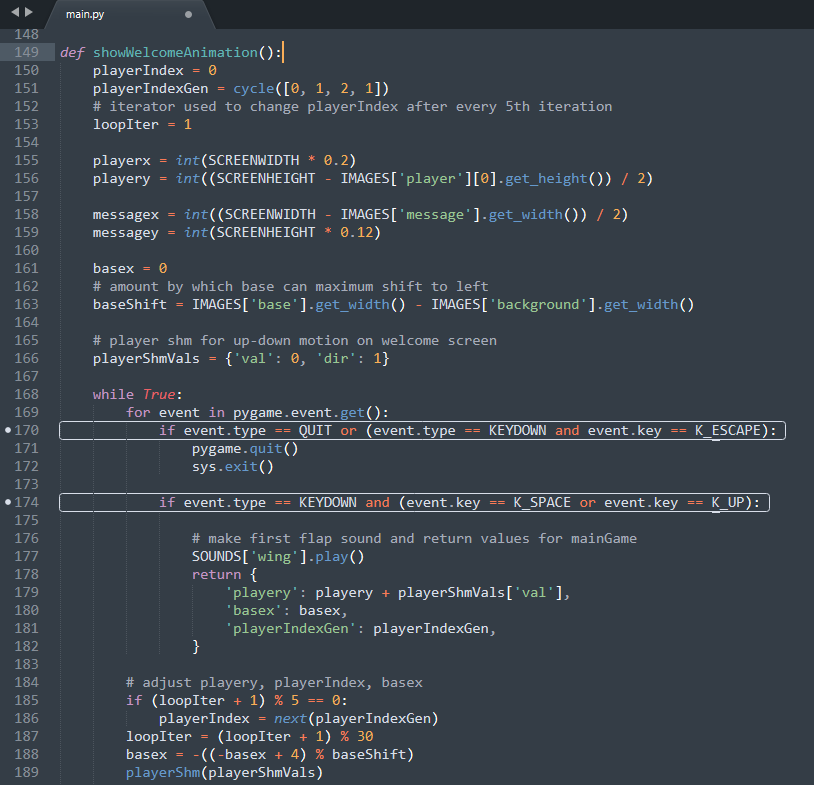
Based on the original Flappy Bird game, all of our members try to read the sample of the original game and fully understand it for further improvement of the game.

We write the separate codes then send it to one guy to do all the commits in the project.

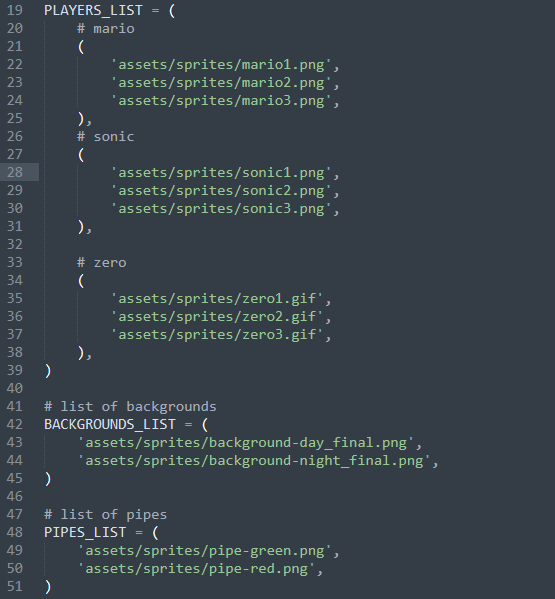
The game menu is first being written. Basically the position of the character appreared with the background and the sentence “Get Ready!” which is also the sprite on the internet.



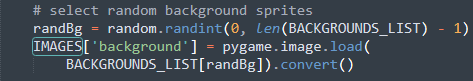
Here is the code for the welcome screen:

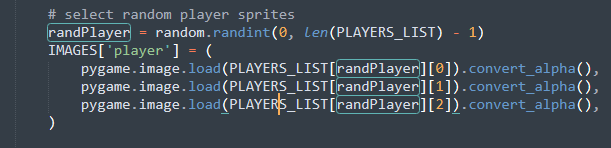


As we can see, first we will choose randomly one character from the list, the list of players and pipes are here:

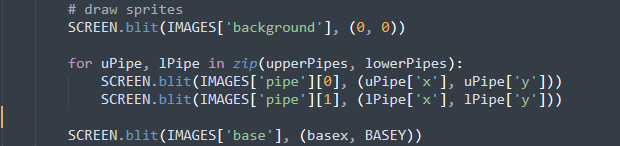


Here is the function to randomly select one character and background

:



Then load it to the game:

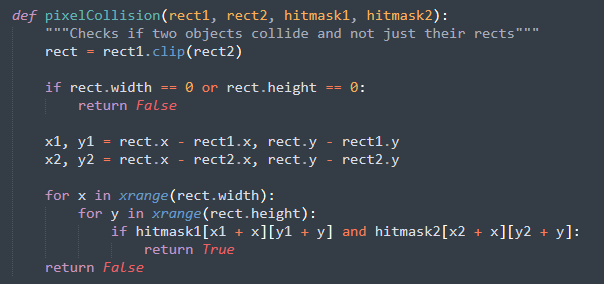


Then It comes to the main() function. In this main function, there are a lot of sub functions to do the small tasks like to check if the player collide with the pipe, to generate random pipes with random length, to get the width and the height of each images, or to increase the score, check if game is over…

Each of them are the collaboration of all team members, we divide it into small tasks and write it, then send all of them to one person to commit to the project.

When the main game starts, it will returns the value of the showWelcomeAnimation() function, which is the position of the character in the Oxy coodirnate system, as we can see in the code is playerx and playery. Then the base will begin to loop and random pipes are generated. When the character cross one pipe, the score will increase by 1. Meanwhile, we have to check if the character collides with the pipe, if It does, the character crash and the function gameover() is called.

Here is the function to get random pipes,show the score, check for collision and check if the character crashed:





So the flow in the program is: Show the welcome screen -> go to the main game -> Generate random pipes -> constantly check if the character cross a pipe or collide with them -> If the character cross, the score increases, if It collides with pipe, the character crash then game over, start again.

We did a little improvement with the character, make it can move around while flapping, so it can saves you if you misclick the flap button at the wrong time, by changing the playery value when the user click Left Arrow or Right Arrow.

1. Specifications
   1. User story

The user story of the game is very simple. The user wants a game like Flappy Bird but not as hard as the original game, maybe with some improvement in gameplay to make it more easy to play. The Flappy Bird game focus in the simplicity so there will be not much things to do.

First of all, the user might want a nice welcome screen, not too complicated. Maybe the character can flap up and down. In the main game, the bird must appear on the screen so the player can play, the sounds, the background and the pipes must appear because it is the basic of the game. When a player died and the new game begins, it should switch randomly between characters in the list. The score should appear in the top and increases after one cross over the pipe. Basically the game can loop forever until the character collide with the pipe and crash.

* 1. Requirements

The requirements is specified with the diagram below: