UNIVERSITY OF TROMSØ

INF-1400: Object Oriented Programming Assignment 3

ahe103

12. april 2016

1 Introduction

This assignment gave the task of creating a game similar to Mayhem, a multiplayer space game where you had to shoot down your opponent to land on platforms to refuel and experienced constant gravity

1.1 Requirements

This assignments requirement for being deemed “Passed” were:

- 2 players with 4 control actions

- at least one obstacle of undefined type in the gamespace

- the players must be able to crash into anything. Obstacles, other players, the gamewalls

- there must be a constant gravity affecting the ships

- a display for score must be supplied, and it shall increase when the player does actions that warrant score increase or reduction

- there must be limitations imposed on the players, such as ; fuel, ammo, health etc.

2 Technical Background

If you're not familiar with cProfiler or pydoc it is recommended to read up on what they do. Other than that there will be use of Pygame sprites which have several useful methods that can be used to simplify the code in several ways.

Pygame spritegroups has an inherent draw method so that you dont have to explicitly blit every item yourself. If the sprites are saved in a spritegroup they are called in turn and order when the draw() method is used, just remember to add the she surface at which they're supposed to be drawn upon.

The groups also can have methods in common across all sprites. You could make a method called “update” and call that, then every sprite in that group will call it's update method

3 Design & Implementation

As required in the assignment tasks it's needed to have every object that are to be drawn on the screen to be a sprite. Therefore every object that the game has to use needs to inherit from pygames sprites. With that follows several benefits such as simplified drawing and movement methods.

Global variables needs to be saved in a specific name “congif.py” but that's quite easy to accommodate for. Then the visualisation of what is needed in the game itself; player controlled ships, obstacles (in this case asteroids), bullets to hit other players with.

It was opted for a pickup style refill mechanic in this game, shoot an asteroid and have a 30% chance of an item spawning. One of several; simple fuel, simple health, simple ammo, combined fuel and ammo, combined health and ammo.

These were the only things that were necessary for the game to perform to specification.

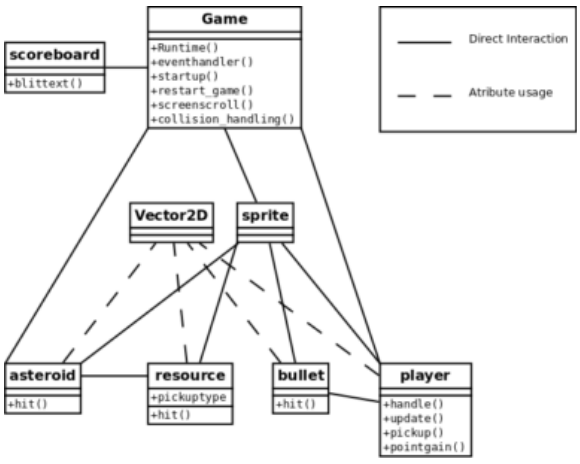

Figure 1: The class diagram Layout

3.1 Game internals

For the game to run it needs a class that creates everything else and sustains itself untill terminated by outside factors like player interrupts or closing the program. The class that sustains itself needs to ready everything for it to play properly, it has to create the players and the obstacles. Check for collisions between everything on the screen, make it move and draw it.

3.2 Game mechanics

The players themselves will create projectiles and adds them to a list for the game to check if it hit any other object that was created. If a bullet then hits an asteroid, the asteroid checks on a random integer between 0-10. If it's lower than 3 then it should spawn a resource of a random choice from a list of resources. The resource will then get picked up when it is run over by the ship

If a player falls off the map he will be quickly damaged until he is destroyed, when he is destroyed the living player is concidered the victor. The points is merely for gaugeing the succes you're having at survival.

The gravity is actually just a constant force to the left at a value of 1 pixel per cycle capped at the max velocity so that they dont exponentialy get dragged more and more

There are some invulnerability frames when you are hit. It's just a coundown from a set value it also alters the appearance of the ship when active

3.3 Animation

There were some elements of the sprites that was embellished upon. Like instead of a static image of a sprite, they moved as if rotating or expelling fumes. This was done rather easily by making a module that took the spritesheet provided and chopped it into predefined images based on the length and height of the image and the number of subimages in the horisontal and vertical direction there was a. it then put those subimages into a list, that list was returned and in turn given to the object that used them. That way when it moved a simple incremental action and performing a modulo operation upon that as to not overindex the list of images created the illusion of movement

The background was done in a different way. The background is simply two images moving to the left, when one image was the negative length of the width of the screen it was set as the positive width of the screen. That way there was no perceivable gap in the background image

4 Discussion

The game I created does not try to look like the game we were supposed to recreate, but I got the go ahead from Marius to create it as I saw fit as long as I implemented all the required features in the assignment requirements. As far as I can see that is true, I have 2 players controlling separate ships, there are obstacles in the “flightpath”. Bullets can hit both players and the obstacles, the players cant escape the confines of the play area unless they fall into “the abyss” and there is constant gravity acting upon the players. All the objects in the play area are sprites

There are some issues with the way collision handling is determined when the ship is checked. It factors in the sprite of the thrust as well as the ship meaning when thrusting with an asteroid behind the player he will be registered as hit. This can be remedied by forcing the check to happen between the ship sprite and the asteroid sprite.

4.1 Evaluation

The results cProfile produced did not show any bottlenecks in the coding, the only thing that took a lot of time was the blitting of sprites upon the game screen. The rest barely made a dent. The results are placed in a .txt file called “cProfile stats” in the same folder as the pydoc and report

5 Conclusion

Although the game I created did not conform to the style of Mayhem it still incorporated most of the mechanics of it. Gravity, Fuel, shooting opponents, obstacles. That and I also learned how to animate something from a sprite sheet although that was not something that was pertinent to the assignment.