Angry Birds Project Plan

Scope of the work:

The programming experience of our group is limited to courses taught in Aalto, and none of us have created a game like this before. We want to create a working game that fulfills all the basic features and at least some of the additional ones depending on time constraints. We want the style and gameplay of our game to resemble that of the actual Angry Birds game.

The idea of the game is that the player tries to kill enemies with different types of birds given. Enemies have 100 hp and in order to get points, the user needs to hit or kill the enemy. Points gained will be equal to the loss of the enemy's hp. We would like to have at least 2 birds with special action and at least one basic bird. The special actions would be speed boost and explosion. Sounds will also be added to our project and the SFML library working for our media playback. SFML will be also used to control the attack with the mouse.

We aim to pursue basic graphics for a start, no animations needed, and the view will follow the bird as it moves sideways. We will be using SFML for our multimedia library. We aim to include a simple user interface that shows information such as points, throwables left, enemies left and hp of each enemy. In the end we can improve them, if we have time left, with some animations and textures.

Players can have ranks according to our game logic. The player gets points from destroying the enemies. Enemies have a certain amount of health and when health is 0 or under, the enemy is dead. Max health is 100. For example, if a user hits the enemy directly with a bird, the enemy dies immediately and the player's score increases the amount that the enemy has lost hp, which in this case is 100. We are planning to create at least 3 different game levels with increasing difficulty. Difficulty is increased by adding more obstacles in the playground and moving their positions. In addition we are focusing on doing different game modes like reaching goals in time or reaching a certain amount of kills with given tries.

Players can enter a nickname, which will be added to a list. In the list you can see every player's scores, passed levels, starts and their names. Stars rate how well a player has succeeded in the levels. 3 stars is max and 1 is minimum. 3 stars can only be achieved if the player manages to kill all enemies with his first shot. The list will be updated when more points are received.

The high-level structure of the software:

Classes

- Abstract class Bird
    - Parameters
        - Name
        - Size
        - Mass
        - Damage
        - HP
        - xPos
        - yPos
    - Functions
        - Static function Sound
        - Static function Special action
- Classes for different birds (inherited from Bird)
    - Functions
        - Special action
        - Sound

The class Bird stores the relevant information for all the birds in the game and it has subclasses for each of the different types of birds. All birds have a size and mass, the base class also has a function for sounds. The subclass houses the information of each bird's special actions and sounds.

- Enemy
    - Parameters
        - Name
        - Size
        - Mass
        - HP
        - xPos
        - yPos
    - Functions
        - isAlive

The enemies, or pigs, use the class Enemy, which stores the relevant information for all the enemies. The class also has a function to tell whether the pig is alive or not.

- Obstacles
    - Parameters
        - Size
        - Mass
        - HP
        - xPos
        - yPos
    - Functions

        - isAlive

The obstacles use the class Obstacles, it has a size, mass, hp and positions. The function isAlive keeps track of whether the obstacle has been destroyed.

- Game (includes movements, collisions etc)
    - Functions
        - Pull
        - Fly
        - Collision
        - BounceBack

The class Game implements the movements of the birds. The function pull calculates the speed and direction of a bird when it leaves the slingshot, fly calculates the trajectory, collision determines if the bird hits something and bounceback how it bounces after a hit.

- Graphics
    - Functions
        - BuildBird
        - MoveView
        - BuildEnemy
        - BuildLevel
            - Places obstacles and enemy
        - NewGame

The Graphics-class does everything related to the graphics of the game, it has functions to create birds, enemies and to build the levels. It also has a function to move the view to follow the flying bird.

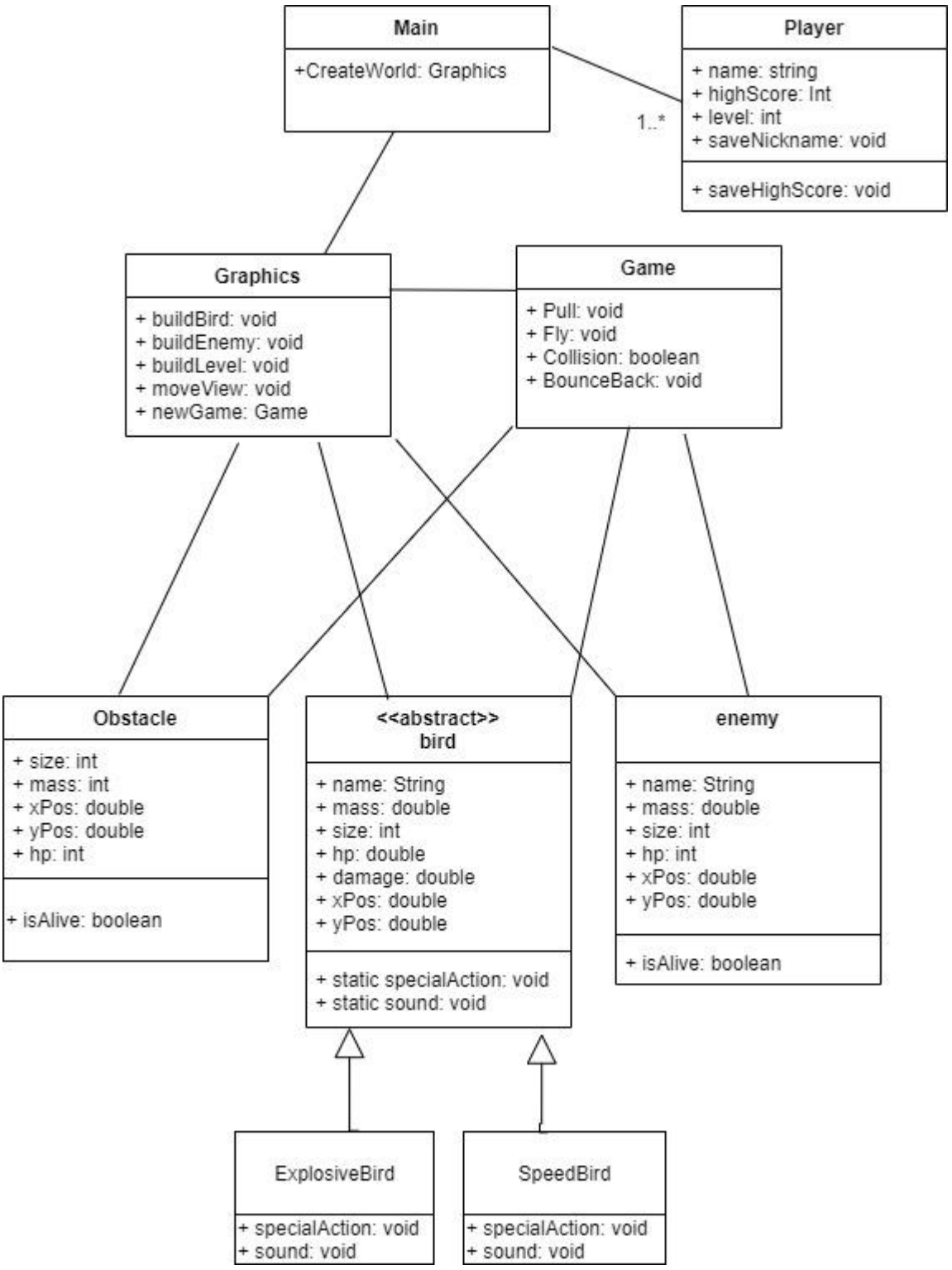
- Player
    - Parameters
        - Name
        - Highscore
        - Level
    - Functions
        - SaveHighScore
        - SaveNickName

The class Player stores information about the player.

- main
    - Functions
        - CreateWorld

Files

- Text File for saving the game data
- Different images for birds and enemies


Main
+CreateWorld: Graphics
Player
+ name: string
+ highScore: Int
+ level: int
+ saveNickname: void
+ saveHighScore: void
1..*
Graphics
+ buildBird: void
+ buildEnemy: void
+ buildLevel: void
+ moveView: void
+ newGame: Game
Game
+ Pull: void
+ Fly: void
+ Collision: boolean
+ BounceBack: void
Obstacle
+ size: int
+ mass: int
+ xPos: double
+ yPos: double
+ hp: int
+ isAlive: boolean
<<abstract>>
bird
+ name: String
+ mass: double
+ size: int
+ hp: double
+ damage: double
+ xPos: double
+ yPos: double
+ static specialAction: void
+ static sound: void
enemy
+ name: String
+ mass: double
+ size: int
+ hp: int
+ xPos: double
+ yPos: double
+ isAlive: boolean
ExplosiveBird
+ specialAction: void
+ sound: void
SpeedBird
+ specialAction: void
+ sound: void

The planned use of external libraries

For our project, we plan to use the SFML library as our multimedia library as it seems to have extensive documentation and tutorials available online. We would like to implement sounds for the game, which can also be done using the SFML.

For the physics of the game, we plan to use Box2D as it was suggested in the project description and it also has extensive resources available. The physics is required for simulating the movement of the birds and the collisions between birds and obstacles or enemies.

As we have not yet started to do any actual coding and we do not have any experience using these libraries these plans may change once we start building the actual game.

Division of work and responsibilities between the group

Everyone is supposed to do the same amount of work. We are dividing all classes into tasks and everyone of us is going to implement the task. When someone is finished with his task he will inform us and we gather around and analyze if the task is functional enough. Then we give him another task to implement or if somebody needs help or is behind a schedule he can help him.

We have estimated time for each task to model how much time each task requires. Sami is responsible for implementing birds and Janne`s job is to work with obstacles and enemies. Rami and Lassi will take care of graphics, sounds and physics.

Planned schedule and milestones before the final deadline of the project

We want to start the coding process at the beginning of the next week and hold a group call on Thursday night to discuss what we have accomplished. Further along the development process, we want to hold weekly sessions to discuss the state of the project and run tests to make sure everything is working correctly. Everyone will do the coding on their own time and if needed, we will hold additional calls to discuss things.

We want to have at least a basic version of the game running by the end of november so we can have a lot of time to polish things and add some of the additional requirements.