Vengeful Pigs

ELEC-A7151/Summer 2020
C++ Project

Meri Mäkelä, meri.makela@aalto.fi
Tuomas Rislakki, tuomas.rislakki@aalto.fi
Aleksi Sohlman,
Henri Välimäki,

1. **Instructions for building and using** the software
    - How to compile the program ('make' should be sufficient), as taken from git repository. If external libraries are needed, describe the requirements here
    - How to use the software: a basic user guide
2. **Testing:** how the different modules in software were tested, description of the methods and outcomes
3. **Work log**
    - Detailed description of division of work and everyone's responsibilities
    - For each week, description of what was done and roughly how many hours were used, for each project member

1.0 Overview

Vengeful Pigs is an Angry Birds clone where the roles have changed. In this version the pigs are after the birds. The birds are shot from a cannon towards birds in their fortifications. There are different kinds of pigs with different abilities. Similarly there are three different kinds of birds. Levels are beaten in the same way as in Angry Birds. When all the enemies are defeated the level ends. The level also ends if there are no pigs left.

The game has three exciting levels in total. The levels are accessed from the main menu. The levels have to be played in order, which means that the second level is not available until the first level has been beaten.

In the main menu the player can also check the highscores of the three levels. The scores are given based on hitting and destroying boxes and birds. Eg. destroying a basic bird gives the player 700 points. The player's performance on a level is rated with stars after beating the level. Highscores are written to a separate file and read when accessing the highscores menu.

2.0 Software structure

The game is run through the game class in which a while loop polls events from the window and then based on the state variable decides which classes it gives the events and calls the update and draw functions for, or if it closes the window. The different classes that the game-class calls are the Level-, Menu-, highscore-, scoreboard- and levelMenu-classes.

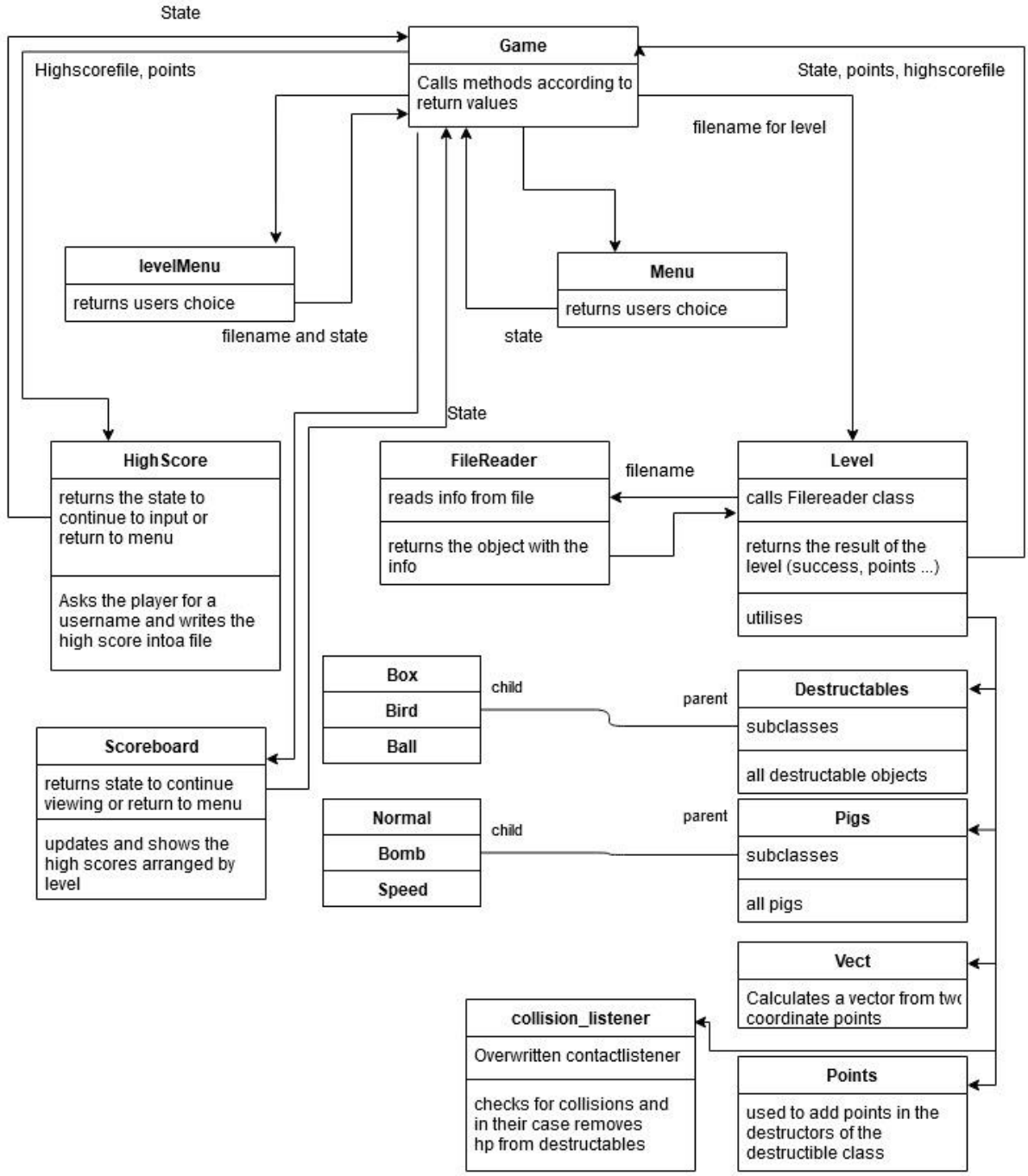
The main menu works as the title screen. From it the player chooses to either go to the scoreboard or level selection.

In the level menu the player chooses which level they decide to play. The class then returns the filepath of the level in addition to the state.

The level class is responsible for running the level and returning the points gotten and if the player succeeded or not. It utilises the Destructibles and Pigs Classes with their child classes to build the level. In managing the cannon it uses the Vect helper class and in counting the points it uses the Points class. The points class is used to add points inside the destructors of the Destructables-objects.

After successfully completing a level the highscores class asks the player for their username and the player either submits a username and the class updates the highscores file or the player exits with esc and the files are not updated.

To add the collision logic to destroy objects the b2contactlistener was used. In the case of a collision it subtracts the amount of the impulses from the hp of the Destructables object.



4.0 Instructions

Setting the software up:

- Make sure you have SFML installed, preferably 2.5, if lower add a line to cmake for findSFML

- Run cmake .. from the build folder
- Run make
- execute the binary

Playing the game:

- Navigating in the menus can be done with the mouse or with the arrow keys. Selecting something happens either with the left mouse or enter. Going back is done with the esc key.
- Inside the level the camera can be scrolled with either the left and right arrow keys. Or by clicking and dragging. Zooming can be done with the mouse wheel.
- You can see remaining pigs and the type of the next pig in top left corner of window
- To fire the pig click inside the red circle and draw out. In front of the cannon forms a red triangle indicating the strength of the shot. While the pig is flying, clicking the mouse activates the special ability of the pig.
- Exiting the level is done automatically when it ends, or it can be done with the esc-key.
- If you get a highscore, the program asks you to enter your name. Save your name by pressing enter.

Known bugs:

- Program may not work on all platforms
- Screen flashes between game and level menu
- Program cannot open files on Mac OS Catalina

5.0 Testing

There are no separate test methods for testing the modules, mostly the modules were quite rigorously tested by the developer after every large change. Additionally after every merge everything was checked and tested. If any problems occurred modules were modified to operate smoothly with the rest of the program.

6.0 Work log

6.1 Division of work

Meri:

- basics for the destructible objects (textures, shapes, physics); classes bird, box, ball (destructibles) - edited by Tuomas later on
- game class + main function
- level menu class
- main menu class
- pictures for bird and pig objects (3+3)
- merging classes together and debugging

Tuomas:

- Level class and any added helper classes
- Filereader class
- cmake to an extent
- Making merges

Henri:

- Pig classes and special abilities for pigs
- Highscores class
- Scoreboard class

6. 2 Weekly workload

Week 29

MERI:
Group meetings about project
Writing the project plan
Time: 7h

TUOMAS:
Group meetings about project
Writing the project plan
Time: 7h

ALEKSI:
Group meetings about project
Getting familiar with SFML

TIME: 7h

HENRI:
Group meetings about project
Getting familiar with Box2D
TIME: 7h

Week 30

MERI:
Setting up SFML
Trying to figure out how to use SFML, Box2d, cmake and git (only SFML was figured out)
Starting to code the destructible classes
Time: 6h

TUOMAS:
Setting up libraries, in the end installed SFML from file and build Box2d from files
some CMAKE research
Time: 10

ALEKSI:
Getting familiar with box2d
TIME: 5h

HENRI:
Setting up SFML
Getting familiar with CMake
TIME: 8h

Week 31

MERI:
Finishing the basic models for classes Bird, Ball and Box.
Creating a test program to try them out.
Searching up SFML and Box2D.
Time: 13h

TUOMAS:
Made the first draft of the Level class
Figured CMAKE, SFML and Box2D out and made an example
Time: 14 h

ALEKSI:
Made some improvements on the the example Tuomas had made and made a draft of the box object class files
Time: 7h

HENRI:
Struggling with CMake and trying to do too fancy things
Basic versions of Pig classes
Making tests with Box2D and SFML together
Time: 15h

Week 32

MERI:
Finally getting Box2D to work
Information search
Time: 3h

TUOMAS:
Starting filereader class
Waiting on Destructables and Pigs
Changed box2d to source files
Time: 3h

HENRI:
Finishing basic features of Pig classes
Learning more about Box2D for special abilities
Time: 6h

Week 33

MERI:
Adding physics to Bird, Ball and Box classes.
Adding new textures.
Creating a test program and trying to run it (and failing).
Trying to figure out cmake (and failing).
Time: 7h

TUOMAS:
Scrapped level class and restarted
Work on file reader
Time: 6h

HENRI:
Making tests with explosions
Figuring out textures
Learning more about git
Time: 3h

Week 34

MERI:
Setting up a game class, main menu class, menu for levels and test programs.
Applying working state logic to game class.
Figuring out how to fit level class to game function calls.
Creating state struct.
Searching through the internet how to get these things to work.
Group meetings.
Time: 29h

TUOMAS:
Making the level, implemented the classes Destructables and Pigs
Creating state struct
Group meetings
Time: 25h

ALEKSI
Group meetings
Getting familiar with Tuomas' level class files
Time: 4h

HENRI:
Implementing special features for pigs
Group meetings
Starting up Highscores class (file reading and writing)
Time: 6h

Week 35

MERI:
Finally figuring out how to make cmake work.
Going through highscore class.
Drawing pictures for birds and pigs.
Merging game and level branch (with Tuomas).
Merging rest of the classes to the "working" part.
Debugging.
Time: ~28h

TUOMAS:
Added the logic to destroying stuff
Added diverse drawfunctions
Added the cannon
Added points
Merging branches together with Meri
Writing documentation
Time: 12 +

ALEKSI
Worked on the visuals of the levels
Added the functionality to move the camera by dragging the screen with mouse
Tweaked some values regarding the feel of the gameplay (cannon power, character hp, etc.)
Designed the levels
Fixed some bugs
Wrote some parts to the documentation
Time: 11h

HENRI:
Learning how to display user input on window
Finishing Highscores class
Making Scoreboard class
Bug fixes
Time: 11h