C++ programming project plan

Hill-side Racing

Paavo Hietala
Veera Itälinna
Oskari Lehtonen
Markus Toivonen

Introduction

For the project of C++ Programming course, we will create our own version of a classical Hills Racing 2D game. We will be using the Box2D physics engine and SFML media library. These are open source libraries that are published with the zlib license. We haven't done anything of substance yet, but luckily there are a lot of tutorials in the internet to help us in the usage of Box2D and SFML. Our main goals are an overall pleasant gaming experience and creating a well structured C++ project.

Gameplay

When the application is started, the user sees a main menu with buttons for (at least) starting the game, viewing the high scores and exiting the program. In the actual gaming mode, the player drives a motorized vehicle through a 2D obstacle course while trying to reach the goal in time. The player receives points for reaching the goal fast and collecting tokens along the way. The game ends when you reach the goal, run out of time or end up stuck upside down.

Should you make it to the top 5, the program will congratulate you, and the score is saved in the “High scores” table. In addition to the vehicle and the track, the user interface of the game shows the amount of time left, current score and the number of tokens collected.

Schedule

The plan is to get the environment set up within a week, and then start creating the obstacles & maps, vehicles, and camera movement. Our plan is to get it working at a very simple level at first, and then start building on top of it. Our group consists of very busy individuals who are active in many fields, so having enough time to do the project might be a challenge. However, we believe in ourselves and our capabilities, and will deliver a working product.

Classes

These are some classes that are probably needed or useful in the game structure. Most likely this will not be the final class structure of the finished game, as our understanding of the problem is not yet perfect.

Game -> for initializing the main window and controlling the state of the game (playing, paused, showing the menu, exiting)

MainMenu -> shows the main menu window and handles the clicks of menu buttons

ObjectManager -> repeatedly iterates through a list of game objects and calls the draw and update methods of each object

GameObject -> virtual functions for updating and drawing objects

Vehicle (inherited from GameObject)

Private: [max speed, tire friction, torque]

Vehicle types (inherited from Vehicle):

Mersu
Ferrari
VW

Obstacle (inherited from GameObject) -> change the attributes of the vehicles upon collision

---

Obstacle types (inherited from Obstacle):

Sand
Stone
Water
Other stuff

Token (inherited from GameObject)

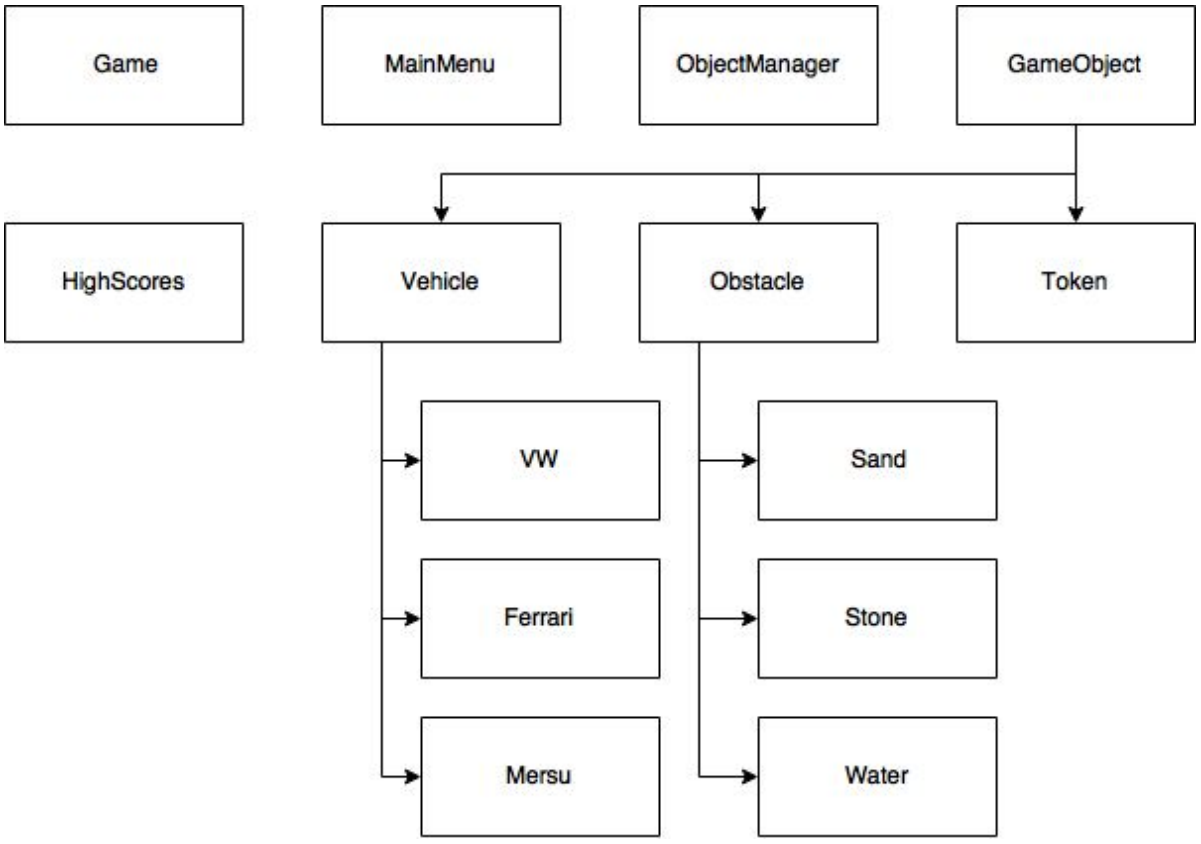
HighScores -> calculates the score

+1 from token collision
+ the remaining time (seconds) from the given time

If a player gets a top five high score the player will get notified and the score will be saved in a text file of top 5 scores.

We will have to implement a collision detection to make the Obstacle class work. Also the SFML and BOX2D libraries provide efficient physics and object-drawing capabilities which will be used to implement different features for the game.

Class hierarchy diagram



Work delegation

We decided main areas of responsibility for each group member. However, all of us will take part in implementing each of these elements.

Oskari - Game physics

Veera - Visuals

Markus - Level designs and testing

Paavo - Movement of the vehicle & screen

Planned extra features

Below are listed some extra features we thought would be relatively easy to implement, while greatly improving the gaming experience.

- Sounds
- Better graphics
- Different vehicles with different dynamics