Christopher Pedersens Minigolf Engine

Minigolf Engine

Written by
Christopher Pedersen

Index

Overview

The minigolf engine contains a number of classes for handling various actions, such as user inputs, drawing items, and reading files. The general architecture is divided into two groups, the data layer, where I keep all objects that are present in the game and primitive elements, and the handling layer that handles actions based on certain situations, such as user input.

An overview of the classes can be seen below:

Handling Layer

- Minigolf.cpp (main class)
- FileReader.cpp
- KeyHandler.cpp
- Render.cpp
- Camera.cpp

Data Structure Layer

- Level.cpp
- Tile.cpp
- Point.cpp
- String.cpp

Explanation

This section contains information about what each class in the architecture does.

On startup the program takes an argument, if you want to test another level, if this is not given, the default level will be chosen.

After this, all of the handler elements are created and relevant pointers are assigned to them. The render function starts and first the camera is updated according to the user inputs, by using the update method on the Camera class, and then drawing methods in the Render class are invoked.

Whenever a key is pressed, the key handler class makes sure that the correct value are passed onto the camera function, that in turn also modifies its own variables to give the correct result.

Minigolf.cpp

This is where the main function is, that handles all the basic glut functions. The class sets up the general glut environment and sets listeners for keys and handles functionality for rendering on the screen.

Although it does handle the actual listeners, most functionality is passed to other classes such as the Render, Camera and the KeyHandler classes.

Lighting is still set up in this class.

FileReader.cpp

The file reader takes a file name as a string and opens the file. When a file has been opened on the object it is possible to pass a pointer to at Level object onto it, and all the data from the file created on the file reader will be pushed into the Level object.

KeyHandler.cpp

All keys that are pressed within the Minigolf.cpp class are passed onto functions in th KeyHandler class. The keyhandler takes a pointer to the Camera class so that it can act on the camera view according to the user inputs.

Render.cpp

The drawing of the Minigolf.cpp class is done by calling functionality placed in the Render class. It contains methods for creating tiles, holes, cups and balls. Since drawing the level requires knowledge and access to it, the Render class takes a pointer of the level as input.

Camera.cpp

The user input so far controls the camera movement. In the Camera class functionality is added so that user interface classes can use methods to invoke movement.

Level.cpp

The Level class contains information about where things are placed. Therefore it contains all the tiles and points on it.

Tile.cpp

This class represents a tile in the game. The tile can have any number of edges if so chosen and it keep all of these edges as a vector list of double vector for the coordinates, x, y and z;

Point.cpp

A point is not a coordinate, but a class that defines objects that are not tiles or balls. So far these elements are only the cup and the tee, but they have similar input structure and are therefore represented by this class, which holds their id and position.

String.cpp

The String class contains slight functionality for handling string elements. So far it handles getting the next elements in a string until a specific character is met. This is only used as part of the FileReader, when getting all of information for the level, from the file.

Considerations

I tried to create most of the base elements I needed in classes, as to make it more manageable and logic at the higher levels when drawing the specific items. This also means that in the beginning there was created a class called Point, as there is now, but this represented a simple point containing three values, for the x, y and z axis. Since I could represent this as a vector of vectors under the tiles, and points so far did not have anything by coordinate values, I removed the class and later created it again under a new meaning.

As a basis I have tried to divide all the functionality into logical named classes so it would be easier to change code and implement new options. However as of now most of the variables and objects are placed in the Minigolf class and references are passed down to other functionalities that need it.

I have chosen not to implement mouse movement of the camera, since I believe that the mouse will later control the direction of the shot for the ball and the power, however it would be possible to switch the controls at some point in the developing process with minor code changes.

The only interesting functionality I implemented would be the String class and the way it searches through strings. I created the next() method because I wanted more easy access to cut up the strings and return relevant values. However the next() method does not seem optimal and might be subject to change at a later date.

Resources

I went through a few tutorials that showed me how to setup things and create certain elements. The resources I used can be seen below:

1. Lighthouse 3d - http://www.lighthouse3d.com/opengl/glut/index.php?1
2. NeHe Productions - http://nehe.gamedev.net/lesson.asp?index=02