Shell Engine Modifications

Advantages

The additions made to the shell engine include a new class called "GameObject", which manages a list of pointers that point to different objects in the game. This is a dynamic list of objects that helps improve cohesion by having the advantage of adding any number of objects to the game shell.

Another implementation is the use of lists instead of flexible arrays, for example when creating a list of GameObjects inside the GameManager class, the user can easily design the level by changing the number of enemy spaceships that appear on the screen along with the number of rows and the number of asteroids. This is more helpful than using an array because this way, the user can later add different game objects. Otherwise, an array would not allow changing the initial size.

An ObjectManager class is also provided, its primary scope being object creation, updating and rendering. By using the list of game objects, the game manager checks for inactive objects in the Update function and deletes them. Moreover, at the end of the game, all objects will be deleted from memory.

Deleting inactive objects is done inside the ObjectManager class by looping through all objects and deleting those that are inactive ("DeleteAll" method). However, a virtual destructor was used in the GameObject superclass in order to decide what to delete at run-time.

Inside the GameObject class, a CheckCollisions() loop is provided in order to detect if a collision happens and check each object to see if they have collided with something. To detect the collision type, each class has a method that returns an identifier (e.g. the Spaceship class is of object type "ship").

Limitations

When checking for collisions, I am using the coded type information, which is a straightforward method, but some human errors were inevitable when dealing with programming it. A more reliable method that could have been used in order to determine the object type is using the "typeid".

The purpose of the ObjectManager class is to prevent the Game class on being dependent on the individual game objects. To further improve the shell engine, an "Object Factory" could be used in order to create new object.

The example game created demonstrates the capability of adding new functions in the object manager. However, in a much larger project, it would not be good practice to create these new functions inside the ObjectManager class. A solution for this could be implementing some sort of component architecture.

References

- Background image: https://i.pinimg.com/originals/8d/bb/58/8dbb588536397d8571c934332ff1dffc.jpg
- AlcWilliam Spaceship Pack: https://alcwilliam.itch.io/alcwilliam-space-ship-pack