Ocean Cleaning Frenzy

Game Design Document

Anjolaoluwa Aina

February 22nd, 2022

Revision 4.0

Table of Contents

General Overview

Ocean Cleaning Frenzy is an action game where players must clean up as much trash thrown from the top of the ocean before the timer runs out.

The Game

Players are given 60 seconds to collect as much trash thrown from the top of the ocean as possible. Any time players can successfully catch the falling trash, they will get 100 points added to their score. Every five seconds, fish will fall along with the trash. The speed at which it falls is faster than the trash, and every time the player catches the fish, they lose 50 points.

Playing a Game

The game starts off with an introductory page which explains how the player can play the game, and the controls that the game supports. After pressing the Enter key, the main game screen is displayed and the game starts. The player will try to collect as much trash as possible by using the left and right arrow keys. Any time the player needs help, they can invoke the help module by pressing ctrl + h, and to return to the game, press the Enter key. Once the timer ends, the player's score is displayed, and they are asked if they would like to play the game again. If they want to, they can by pressing the Enter keyboard. At any point during the game, the player can terminate the game by pressing the ESC key.

Target System and Game Engine

The game is targeted towards PC users that have Windows and was developed using the Allegro library. All the sprites and bitmaps used by the game were created in Photoshop.

Formal Requirements

The requirements for the game are listed as follows, from the most important requirements to the least important requirements.

i. The game must contain at least two sprites, where at least one sprite is drawn and used in-game.

ii. The game must contain a diver sprite to be controlled by a player using the left and right arrow keys.

iii. The diver sprite must contain two animation frames representing the sprite's directions in game, in which the sprite is facing left, or right.

iv. The game must contain sprites representing trash, and fish.

v. The game must begin with an introductory page that tells the player what the game is.

vi. The game must have a visible timer to inform a player on how much time they have left until the game concludes.

vii. The game must display how much points the player has on the bottom left-hand corner of the game screen.

viii. A player must be able to terminate the game at any point in time by using the ESC key.

ix. A player must gain 100 points every time the player collects trash.

x. A player must lose 50 points every time the player collects fish.

xi. The game must have a simple help module that can be invoked by pressing ctrl-h.

xii. The game must have background music that can be turned on and off by pressing ctrl-m.

xiii. The game must have an attractive interface.

xiv. The game must include some kind of multithreading aspect (i.e., use more than one thread in the game to execute some kind of task – such as updating the screen)

The Sprite Class

The Sprite class contains all the methods and instance variables related to a sprite. It contains instance variables such as the width and height of the sprite, the x and y position of the sprite, the image of the sprite, and whether the sprite is alive or not. Some of the methods included in the class are the load method, which loads a BMP file into the image variable, the draw method, which draws the current sprite to the specified destination, methods that return the x and y position of the sprite, and set the x and y position of the sprites, methods that get and set the alive state of the sprite, and a collided method that returns true if a position has occurred by comparing the boundary rectangle of the sprite to the boundary rectangle of another sprite, and false if no collision has occurred. This collision method is very important as it helps to determine whether the diver sprite controlled by the player has collided with another sprite in the game.

The Player Class

The diver sprite is the most important sprite used in the game and is controlled by the player. The Player class contains the diver sprite as an array of Sprites, and has two other instance variables, which are the direction the player's diver sprite is pointing at (left or right), and the score of the player. There are setters and getters to return the score, and the changing frame of the diver sprite are all handled in the Player class rather than the game class, to cut down on the amount of code already present in the game class. When the player presses the left arrow key,

the direction of the player is set to left, and the diver sprite frame is changed, and updated. When the player presses the right arrow key, the direction of the player is set to right, and the diver sprite frame is changed, and updated.

The Fish Class

The Fish class is very similar to the Player class but does not contain an array of Sprites. Instead, it contains a single Sprite that represents the fish. It is the only instance variable in the class, but there are a few methods contained in the fish class, such as the move_fish method, which moves the fish down, the handle_fish_out_of_bounds method, which returns the fish to the top of the screen (granted that the fish is still alive), the get_fish_sprite method which retusn the instance variable, and the respawn_fish function that makes the fish alive if it is not alive anymore. The reset_fish method is only used when the player wants to play the game again, which resets the fish back to its default values. There is a private method in this class, but it is only used to generate a random x position every time the fish reaches the bottom of the screen, or the fish has collided with the player.

The Trash_Pool Class

The Trash_Pool class is very similar to the Fish class but contains an array of Sprites, where all the different trash are contained. It is different from the player's array of Sprites because each image in the array are different sprites, while the array in the Player class contains the same diver sprite, but either facing left and right. It is also the only instance variable in the class. The methods that are contained in the class are methods that load the sprite images into the array of Sprites, getting a sprite at a specific index, spawning a sprite,

moving all alive sprites, resetting the position of an alive sprite, and resetting the trash pool. There also contains a private method in the class used to generate a random x position every time an alive trash reaches the bottom of the screen or has collided with the player.

The Game Class

The Game class controls the execution of the game. It has many instance variables such as the background of the game, the music used in the game, a Player, Fish and Trash_Pool object, and other game-related data. There are many methods in the game class, but the method that is the most important is the start_game method, which initializes the game by loading all the backgrounds, sprites, fonts and sounds. It calls the initialize_game method to handle it. If the initialize_game method returns false, an error has occurred and are returned out of the start_game method. Otherwise, the background music starts playing and the new_game method starts a new game.

User Interface Screens

The game screen for Ocean Frenzy have been divided into two sections. The top section will contain the game banner, which has the game title and instructions on how to invoke the help module or toggle the background music. The timer and score are displayed on the bottom of the game screen. The game screen contains the area in which the game occurs, including the player’s sprite, the trash and fish sprites.

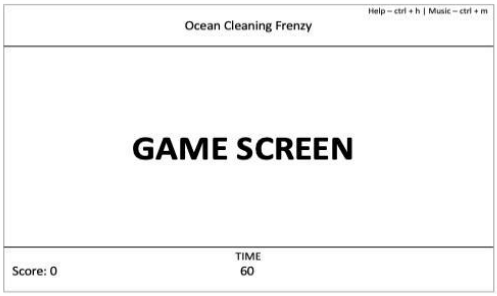

Figure – Main Game Screen

Theme

In this section, the graphics that will be used in the game are explored in the “Graphics” section. Any sound that will be used in the game, including background music and sounds, are explored in the “Sound” section.

Graphics

The graphics used in the game were created to give the game a fun 2D feel to it. Along with that, as the game is set in an ocean, the provided graphics make the player feel as though their character is in an ocean. The background used in the game was provided by Freepik. No attribution is required, as a premium Freepik account was used to download the asset, but has been provided in the ReadMe file provided with the game, nonetheless. It was pixelated using an online image editor. The fish and trash sprites were also collected from Freepik. The diver sprite and its various frames was created in Photoshop.

Sound

The background music used in the game was upbeat and fun. It was retrieved from Pixabay, which also did not require any attribution, but has been provided in the README file nonetheless. All the other sound effects such as when the player collects a fish or trash were also retrieved from Pixabay.

Menus

Listed in this section are the most important menus needed for the game to work. They are the introductory menu, the help module menu, and the end of game menu.

Introductory Menu

This menu explains to the player how to play the game and introduces the controls that can be used in the game. It does not make use of the text out functions provided in Allegro but uses its own custom background that was created for the game, called the “start background”. It is very similar to the game screen and uses the same background but does not contain the timer and score at the bottom, or the help module and music information. The title is also much larger.

Help Module Menu

This menu helps users understand how to play the game. It explains the controls that the player has available to them. The help module has its own custom background displaying this information.

End of Game Menu

This menu displays a player's score at the end of the game and asks them if they would like to play the game again. If they press the Enter key, the game will restart. If they press the ESC key, all bitmaps will be destroyed, and the game will close.

New Additions

Multithreading

The game makes use of multithreading by using the Pthreads library. Multithreading occurs when the game needs to acquire and update the screen, and when the fish spawns every six seconds. This new addition was added by creating static functions inside of the game class and invoking them (creating them) when needed and joining them when not needed. No new class variables were created for the thread functions, nor were mutexes added as the threads do not share any data due to them being static functions.

Using Allegro Datafiles

All of the game resources used in the game are stored using Allegro datafiles. The new file, called game_data.dat stores the bitmaps, fonts and sounds associated with the game. Instead of using the pathname to locate the resource, the datafile is loaded into the game, and objects are grabbed from the datafile by using the pre-defined object names in the datafile.h header file. The game still checks to ensure that all game objects have been loaded into the game properly to avoid errors. The dat application, along with the allegro42.dll are located in the folder of the game, where the headers and source files are all located.

Artificial Intelligence

The game makes use of artificial intelligence by making the fish more smarter. A tracking algorithm was implemented so that when the player moves to the left, the fish also

moves to the left. The fish is now slower and the player must strategically move to get past the fish. It is now harder for the player to collect trash objects without collecting the fish, increasing the difficulty of the game, and making it more interesting. Along with that, the trash do not fall down vertically, but rather diagonally to make them feel more animated, and ultimately improve the overall experience of the game.

Conclusion

This game design document describes what Ocean Cleaning Frenzy is, what a player should expect when playing the game, the functionality required for the game to work, a look at some of the classes used in the game, the requirements that must be added into the game, the screens, and the menus in the game. The document also addresses the new features added into the game. Ocean Cleaning Frenzy aims to show the importance of keeping oceans clean while providing a fun and interactive gameplay.