Sea Skipper Technical Design Document

# Technical Project Description

**Sea Skipper** is an endless side-scroller game with an underwater setting.

The player character is a penguin, and their goal is to get as far as possible in a single run, while also collecting coins and avoiding obstacles such as **Sharks** **Seals** and **Barnacles**, which all will spawn consistently throughout the duration of a single run, getting more difficult to avoid as the run continues as the player will go faster and faster.

# Mechanics

## **Player Movement**

The player movement is one of the simplest mechanics implemented in this game.

The player will move up when the player either left clicks, or tap and holds the screen (mobile/android) and will go down when the player stops giving input.

Luckily, with Unity, just using the ‘**If (Input.GetMouseButton(1)) {}**’ works for both the mouse clicking AND the touchscreen input for mobile.

## **Collision Handling**

Collision checking will be handled by primarily using Trigger Colliders.

The player has a capsule collider attached to it, with the ‘**Player**’ tag.

* **Coins**

Coins use a trigger collider, which is used to check every frame (if the player is still running) if it collides with something. If it DOES collide with something, and what it is colliding with has the ‘**Player**’ tag, that coin game object will be destroyed, and the players collected amount of coins will increase by 1.

Other obstacles, such as the **Barnacle**, **Shark**, and **Seal** have their own individual trigger colliders and use a ‘**Creature Handler**’ script, which checks if it collides with something with the “**Player**” tag, in which case, the players current run will end, causing a game over.  
Otherwise, if they collide with an object with the “**Despawn**” tag, their game object will be destroyed.

## **Artificial Intelligence (AI)**

### **Aquatics**

#### **Sharks**

* **Sharks** are large and will move slowly when far from the player. However, once close enough to the player, the shark will start to move 35% faster until it either collides with the player, or goes offscreen. Every few seconds they have a chance to move up or down for an extremely short period of time, though, still watch out! The sudden movement may catch the player off guard.

#### **Seals**

* **Seals** come a bit bigger than sharks here, though they don’t speed up. Instead, when a **Seal** is about to come onto screen, the player will be given a small warning indicator on the side of the screen that follows the players Y position for ~4.5 seconds before disappearing and the **Seal** appearing. The **Seal** will fly across the screen in a straight line from the position the warning was before it disappeared.

## Flowcharts & Explanations

### Collision Handling

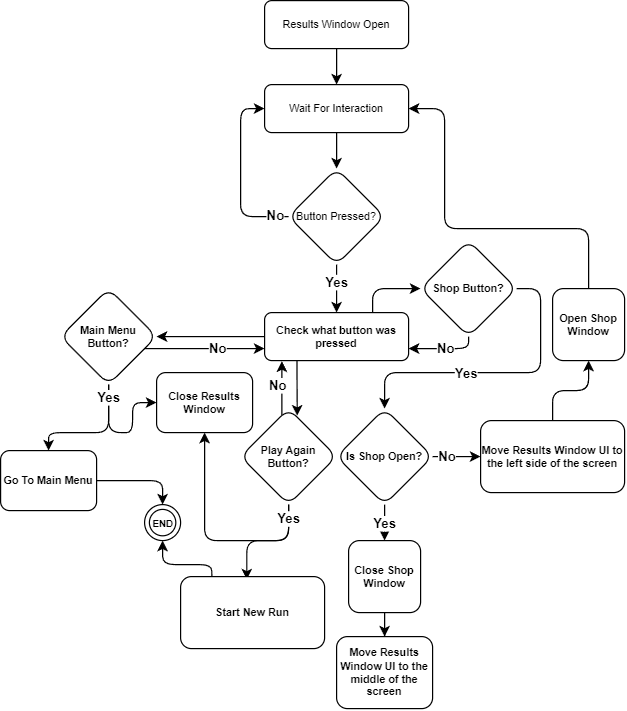
This flowchart starts to apply when the player starts doing a run.

During a run, if the player is colliding with an object, get the object we are colliding with and check its tag.

If its tag is “Obstacle”, ‘Defeat’ the player, making them fly a few extra meters before stopping, then add the amount of coins collected in that run onto the total amount of coins the player already owns, then, start saving the updated information. In the save file, override the previously saved total amount of coins with the new total, then if the player does not have a ‘Best Run’ score, or the distance traveled is more than the currently saved ‘Best Run’ score, save the distance traveled as the new ‘Best Run’.

—---------------------------------------------------------|

### Results Menu



This flowchart applies whenever the Results Window is open.

While the Result window is open, we wait for an interaction with any of the buttons.

If a button is pressed, we check which button it was, then act accordingly:

**Main Menu**

When pressed, we close the Results Window, and change back to the Main Menu.

**Play Again**

When pressed, we close the Results window, then immediately begin a new run.

**Shop**

When this button is pressed while the shop is not open, the Results Window will slide to the left of the screen, and the shop will open up beside the Results Window. Pressing the shop button while the shop is open will close it.

# Assets Required

## Code Assets

### Scripts

* Player controller - player movement, distance calculation
* Game Manager
* Creature Controller (for Shark spawning & behavior)
* Warning Behaviour (for Seal warning indicator & seal functionality)
* Ui Managers (in run) - controlling the distance traveled and coins collected ui
* Ui Managers (end of run) - displaying correct distance traveled and coins collected

## Art Assets

### 3D Models

* Player Model (Penguin)
* Shark Model
* Seal Model
* Barnacle Rock Model

### Sprites & Textures

* Background Texture (underwater + fish)
* Coin Sprite
* Warning Indicator Sprite

## Sound Assets

# Developer Roles

## Emmanuel Williams

* Lead Programmer & Team Leader

### Worked / Working on

* Player Movement
* Collision Detection
* Score / Distance Calculation
* Menu Functionality
* Art (Player Model, Barnacle Rock Model & Textures/Materials, Background Texture + Fish, Coins)
* Prototype Level Design
* Bug Fixing

## Jaxon Haldane

* Senior Programmer & Ai Developer

### Worked / Working on

* Ai scripting & functionality (Sharks, Seals)
* Menu Design
* Model Creation (Shark Model, Seal Model)
* Texture & Sprite creation (Warning Indicator)

## Dominic Sarbu

* Senior Programmer & Level Generation Developer

### Worked / Working on

* Procedural Generation (Level Generation)
* Procedural Generation (Reworking Obstacle & Creature Spawning Mechanics)
* Procedural Generation (Spawn timings & Requirements)
* Bug Fixing

# Production & Testing Schedule

Week 1 30/08/23 - 01/09/23

Brainstorming & Designing + Documentation

Week 2 06/09/23 - 08/09/23

Documentation, Scripting (creating player controller, basic player movement) + Scene Setup (lighting, camera perspective, player character prefab setup)

Week 3 13/09/23 - 15/09/23

Scripting (Updated Player Movement, Prototype Functionality for Sharks - Spawning, Despawning, Movement -, Game Manager - Game Data Saving & Loading -, Player Distance Calculation + Coin Collection during a run, Menu Functionality), Basic Menu Setups (Setting up the layout of the Main Menu, and the Run Ended menu), Art Creation (Player Model, Background Texture, Coin Sprite), Deploy Prototype for first bug testing & feedback, game pitch / presentation.

Week 4 20/09/23 - 22/09/23

Bug Fixing, Scripting (Seal & Warning Indicator Mechanics - Spawning, Collision, Despawning -, Barnacle Rock Functionality - spawning, collision, despawning -), Documentation Updates

Week 5 27/09/23 - 29/09/23

Bug Fixing, Model Updates, Polishing.

# Strategies for Monitoring Production Progress

Using Websites such as Trello for bug testing and feedback

Using Source Control (GitHub)

Team Communication (Discord)