ASSIST
Innovative Minds

# 2D Endless Runner

Scene name: LevelScene (Assets/Scenes/LevelScene)

This document will contain all the features that need to be implemented or fixed in the 2D Endless Runner game.

Notes: The game uses scriptable objects to maintain the data that vary from the data that doesn’t vary. Every level has a scriptable object “LevelData” composed from four other scriptable objects:

- LevelEnemyData scriptable object
- LevelObstaclesData scriptable object
- LevelPlayerData scriptable object
- LevelTrackData scriptable object

Before starting the test, you should check them out:

=> Scriptable object classes: Assets/Scripts/ScriptableObjects/

=> Scriptable object instances: Assets/Data/

# 1. Features that need to be fixed

Main functionalities will be added in the specified functions, but there is no restriction to create another variables or functions that will be used, in order to construct the solution.

## 1.1 Player Controller (Raycast methods)

### 1.1.1 On Ground functionality

=> Implement functionality for detecting if the player is on ground

=> Script: Assets/Scripts/Controllers/Player Controller/PlayerController.cs: line 92

## 1.2 Obstacle Generation (Instantiation methods)

### 1.2.1 Spawn rate of the Crate/HG Zone

=> Implement functionality for setting the spawn rate of the obstacles

=> Script: Assets/Scripts/Managers/ObstacleGeneration.cs: line 49, 56

### 1.2.2 Generation of the Crate/HG Zone

=> Implement functionality for generating obstacles

=> Script: Assets/Scripts/Managers/ObstacleGeneration.cs: line 63, 71

## 1.3 Enemy Generation (Instantiation methods)

### 1.3.1 Spawn rate of the enemies

=> Implement functionality for setting spawn rate of the enemies

=> Script: Assets/Scripts/Managers/EnemyGeneration.cs: line 33

### 1.3.2 Generate enemies

=> Implement functionality for generating enemies

=> Script: Assets/Scripts/Managers/EnemyGeneration.cs: line 55

## 1.4 Enemy Controller (2D Collision)

### 1.4.1 Position of the enemy

=> Implement functionality for updating the position of the enemy

=> Script: Assets/Scripts/Controllers/Enemy Controller/EnemyController.cs: line 47

### 1.4.2 Crate/Death limit zone interaction

=> Implement functionality for crate/death limit collisions

=> Script: Assets/Scripts/Controllers/Enemy Controller/EnemyController.cs: line 56

### 1.4.3 Random jump

=> Implement functionality for random jump

=> Script: Assets/Scripts/Controllers/Enemy Controller/EnemyController.cs: line 68

## 1.5 Weapon Firing (Responsive UI)

### 1.5.1 Gun Reticle position update

=> Implement functionality for updating the position of the reticle

=> Script: Assets/Scripts/Controllers/Player Controller/WeaponFiring.cs: line 151

### 1.5.2 Handle reticle “AFK” state

=> Implement functionality for disabling reticle after a specified time

=> Script: Assets/Scripts/Controllers/Player Controller/ WeaponFiring.cs: line 160

## 1.6 Camera Controller (Post-Processing Effects)

### 1.6.1 Vignette camera effect

=> Implement functionality for vignette camera effect

=> Script: Assets/Scripts/Controllers/Camera Controller/CameraController.cs: line 62

### 1.6.2 Chromatic Aberration camera effect

=> Implement functionality for chromatic aberration camera effect

=> Script: Assets/Scripts/Controllers/Camera Controller/CameraController.cs: line 71, 80

# 2. New Features

## 2.1 Collectable Generation

### 2.1.1 Create Collectables level data SO. Create a new scriptable object with specific collectable object information, similar with obstacles level data SO. Parameters:

=> collectable type (bonus score coin, energy bottle)

=> spawn rate of the collectable

=> prioritization of the collectable spawning (see the game presentation document for more information about prioritization meaning)

=> you can use any assets you want for the visual part

### 2.1.2 Update level data SO. The LevelData scriptable object will need an update for the collectable data.

=> the current level data scriptable object class needs to be update with the information of collectable level data.

### 2.1.3 Generate bonus score coin. Dynamic generation of this type of collectable.

=> the spawning of the bonus score coin is up to you (in air, on the ground, bouncing on the ground, etc.). Also, you can add visual assets for this type of collectable.

=> the spawning should be constrained by the parameters from collectable level data scriptable object.

### 2.1.4 Generate shield stamina energy bottle. Dynamic generation of this type of collectable.

=> the spawning of the energy bottle is up to you (in air, on the ground, bouncing on the ground, etc.). Also, you can add visual assets for this type of collectable.

=> the spawning should be constrained by the parameters from collectable level data scriptable object.

## 2.2 Collectable Controller

### 2.2.1 On Trigger Interaction. Implement the interaction between the player and the collectable objects.

=> bonus score coin interaction: bonus score added to the player score.

=> energy bottle interaction: part of “Shield Energy” feature. This should refill the energy of the shield with a specific percentage, up to you.

## 2.3 Shield Energy

### 2.3.1 Shield functionality. Implement the shield functionality for the shield energy component and other corresponding functionalities.

=> implement separated functionality for the actual shield implementation, that will manage, modify the energy of the shield.

=> the shield should have an energy fill from 0 – 100%

=> energy bottle interaction: part of “Shield Energy” feature.

### 2.3.2 Shield UI. Add shield UI so that the user can see how much Shield stamina he/she has available.

=> implement user interface that will display information about the energy shield (bar progress, text, animation, etc.)

## 2.4 Main Menu UI

**2.4.1 UI for game start.** Add UI (user interface) so that the level scene does not immediately start when you open the app.

**2.4.2 Dynamic UI for available levels**. For this feature you should populate UI based on how many levels you have available. (setup levels from Game Manager script)

**2.4.3 Normal/Endless mode UI**. Add the possibility for the user to select if the play mode should be normal or endless. In normal mode the level end when you reach a certain number of points, the endless mode should be played as long the player is alive.

## 2.5 New levels

**2.5.1 New levels Creation**. Add at least two new levels to the game using whatever resources you want.