Department of Computer Engineering

Games and Multimedia – Games Programming

Teachers: Luís Monteiro, Ricardo Antunes and Roberto Ribeiro

Project Description – First Examination Period – 1st Project

Estimated date to publish the grades: 25/05/2021

Objectives

For this assignment it is intended to develop a platformer game using the **Unity** game engine and the **C#** programming language. During the development of this assignment, the best practices of programming should be applied. This assignment focuses on the development of the enemy, **giving priority to composition over monolithic solutions or inheritance**.

Functionalities to Implement

All the necessary assets to develop the requested functionalities were provided by Unity's 2D Game Kit and are available for download in the **Moodle** platform. A project with the base map is available as well. **The work must be developed using the supplied project**.

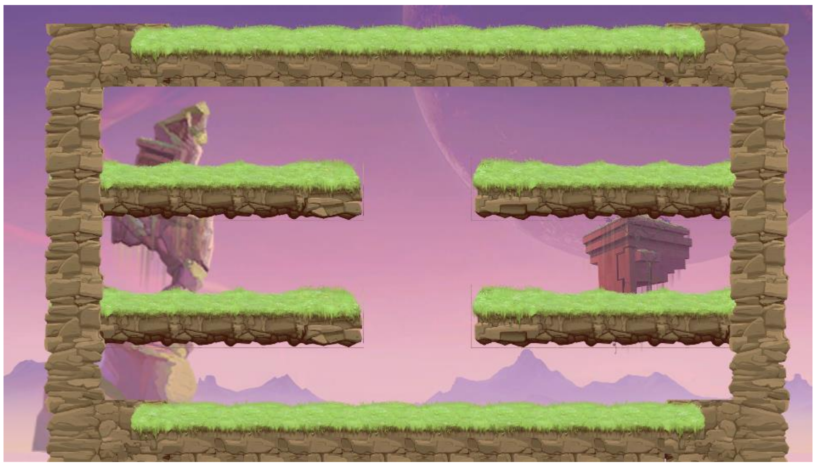

Figure 1. Representation of the playable area

Enemies

The visual representation of the **Enemy** should be constituted by all the animation states required by the project, which are idle, walk, jump, attack and death. For the jump animation of the enemy, each group may use the most appropriate sprites from the available resources.

All the configurable values of the **Enemy** should be available in the **Inspector** (e.g., jump strength, walk speed, bite damage, time intervals).

1. Patrol

The **Enemies** patrol by walking in the direction they are facing. If the **Enemy** sees a wall during the patrol, the movement should be interrupted, the character should turn around and proceed with its movement in the opposite way. The decision of turning around when arriving at the edge of the platform or falling off depends on a predefined interval. The **Enemies** should also detect platforms above their heads and jump to the detected platform. The horizontal movement should be maintained while jumping. The detection should only occur after a random interval between a minimum and a maximum time value.

2. Biter

From the **Enemies** available in the scene, one of them should be elected as a **Biter**. When a new **Biter** is elected, the previous **Biters** should return to their normal behavior and the **Enemies** that already knew who the **Biters** were should forget about it. A new election should be performed after a predefined interval.

When a **Biter** gets close to another **Enemy**, it bites to inflict damage. The **Biter** should only inflict damage if the **Enemy** is near its mouth in the exact moment of the bite.

The **Biter** should be distinguished from the other **Enemies** with another tint color.

Figure 1. Representation of the enemy during a bite

3. Social System

When a non-**Biter Enemy** gets near one of its kind, they should stop patrolling. They greet and if at least one of them knows at least one **Biter**, the one with that information tells it to the other. The **Enemy** that received that information should stay aware of the **Biters**. When two **Enemies** meet and none of them knows about any **Biters**, they can rest (i.e., stay in idle) during a predefined interval and then flip and proceed. If an **Enemy** is bitten twice, it becomes a **Biter**. When two **Biters** meet, they ignore each other.

4. Alert System

When **Enemies** are informed about the identity of the **Biters**, they should become aware of their presence and every time they detect one of those **Biters** nearby, they should run away by turning around, increase the walking speed, and decrease the interval between jumps. After a predefined time running away, they return to the normal patrol behavior.

5. Health and Armor

To fulfil this exercise, add an **Armor System** *script* to the Enemies to protect them from bites. When the **Health System** receives damage, and if an **Armor System** exists, it notifies the latter about the damage. This system should notify back the **Health System** about the remaining damage and be deactivated if the armor points reach 0.

The armor is represented through a diegetic UI element that corresponds to a colored light around the **Enemies'** head. The light changes color when the armor protection decreases, and when there is no armor protection left, it should be removed. This color change should be performed through a **Gradient**.

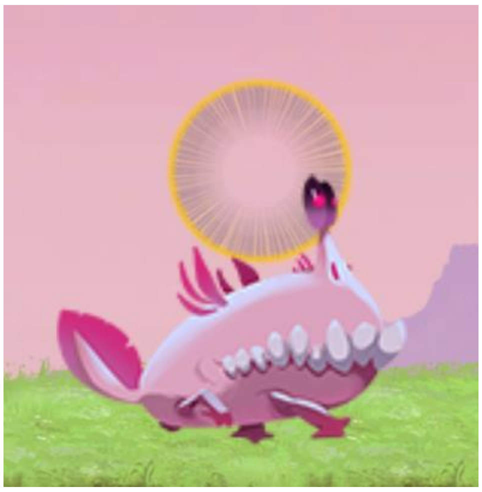

Figure 2. Light representing an armor system with full protection.

6. Report

Besides the project, you must also deliver a report, **in PDF format**, with the description of the implemented algorithms and an explanation for the taken decisions. The algorithms taught during the classes **should not** be described. The report should cite all the bibliographic references (books, websites, etc.) in which the code of the developed algorithms was based.

In the report, the list of the unfinished functionalities (partially or totally) should be **explicitly** mentioned as well. You should not include any source code in the report.

The report should follow the supplied template or, at least, its structure.

Criteria

The evaluation criteria for this assignment are the following.

Functionality	Score
Locomotion	10%
Biter Related Features	40%
Social System	20%
Alert System	10%
Health and Armor	10%
Report	10%

Delivery Rules

The rules listed next must be respected for the project to be evaluated. Otherwise, **the work won't be accepted**.

1. Delivery deadline of the 1st project: **6/05/2021.** The projects delivered after the deadline won't be accepted.

2. The project must be developed by a group of 2 students attending to the same practical shift (the projects developed by groups of more than 2 students won't be accepted).
3. The work should be developed using the provided project available in the **moodle** platform.
4. The developed work (Unity project + report) should be delivered through the **moodle** platform, using the link available for that purpose. **The project must not contain compilation errors or the implementation of non-requested features according to the assignment**.
5. **The report should be delivered in pdf format**.
6. All students must defend the delivered project so that they may be evaluated. The date of these discussions is indicated in the evaluation calendar of the course (These will take place in the week following the delivery date of the project).
7. The project must run in the version **2020.2** of Unity's editor. Every project will be evaluated in this version of the editor as is.
8. The delivery of the project folder **must exclude** the folder "**Library**".
9. The name of the delivered file must be composed by the students' names (first name and surname) and numbers of students who carried out the project, respecting the following structure:

"**NameSurname1_Number1_ NameSurname2_Number2.(ZIP or RAR)**"