



Unity Piscine - Module02

2D environment, tiles and sprites

Summary: This document contains the Module02 subject for the Unity Piscine.

Version: 2

Contents

I	Instructions	2
II	Day-Specific Rules	3
III	Foreword	4
IV	Exercise 00: I See the World in 2D	5
V	Exercise 01: White Walkers	7
VI	Exercise 02: Arms!	9
VII	Submission and Peer Evaluation	11

Chapter I

Instructions

- If you have trouble installing the required tools for your project on the 42 computers, you may use a virtual machine. In this case, you must:
 - Install the virtual machine software on your computer.
 - Install the operating system of your choice.
 - Install the necessary tools for your project.
 - Ensure that you have enough space on your session to install all of it;
 - Have everything installed before the evaluation.
- Only this page will serve as reference. Do not rely on rumors.
- Carefully read the entire document before starting.
- Your exercises will be evaluated by your fellow piscine participants.
- This document is your reference. Do not blindly trust demos or example pictures, which may include unnecessary additions.
- Got a question? Ask the peer to your right. If not, try the one on your left.
- By Odin, by Thor! Use your brain!!!



Intra shows the date and time when your repositories close. This also marks the beginning of the peer-evaluation period for that piscine day. The peer-evaluation lasts exactly 24 hours. After that, any missing evaluations will be scored as 0.

Chapter II

Day-Specific Rules



Module 02 is important for the next module. Here, you'll create the basic elements that will be useful in Module03.

Chapter III

Foreword


Today, you'll start creating the basic elements of your future [game](#).

To do this, go to the [Unity Asset Store](#) and import the Free Pixel Art Overworld Tileset.

In this module, we recommend using a **GameManager** and starting to work with **Tags**. These will be very useful in this and future modules.

Chapter IV

Exercise 00: I See the World in 2D

	Exercise :
Turn-in directory : <code>unityModule02</code>	
Required elements : "Map1" scene and anything useful	
Forbidden functions : None	

To start, create a new Module02 2D project.

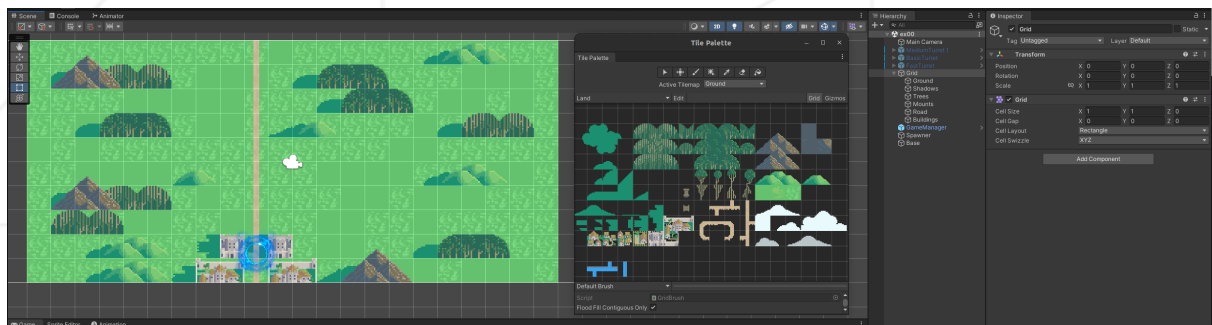
So! With the new tiles you've imported, you'll begin by creating a **Tile Palette**, which will be very useful.

Your palette should be saved in a new folder called `TilesPalettes` inside your `Assets` folder. Make sure to choose a consistent name.

Now that you have your **palette**, you can start creating your map.
Your map must contain:

- The ground.
- The road.
- Trees, mountains, or anything else you want for decoration.

It should look something like this:






For the first map, your road must be very simple. This is the road on which the enemies will move.

Chapter V

Exercise 01: White Walkers

	Exercise :
Turn-in directory : <code>unityModule02</code>	
Required elements : "Map1" scene and anything useful	
Forbidden functions : None	

Now we need enemies!

Since animation isn't our focus today, you'll create very simple enemies using a basic sprite.

A plain white capsule will do just fine. You must have only one script for all your enemies.

You'll also need to create a spawner. Enemies should spawn at the top of the map and move toward the bottom.

For now, keep it simple: use a fixed delay between spawns.

Place your base (represented by a circle) at the bottom of the map. When an enemy reaches the base, it loses 1 HP.

Your base should start with 5 HP.

Each time an enemy reaches the base, print the remaining HP to the console.

When your base reaches 0 HP:

- Display "Game Over" in the console.
- Stop spawning new enemies.
- Destroy all remaining enemies on the map.

In any case, an enemy must always be destroyed when it:


- Reaches the base,
- Leaves the map,
- Is killed,
- Or if the game is over.



- **Spawner:** In video games, a spawner is an object in a simulated game world that generates other objects.
- **HP:** HP stands for Health Points - the total amount of life an entity has.

Chapter VI

Exercise 02: Arms!

	Exercise :
Turn-in directory : <code>unityModule02</code>	
Required elements : "Map1" scene, <code>EnemyController</code> script, <code>TurretController</code> script and anything useful	
Forbidden functions : <code>None</code>	

Organize your defenses!

In the assets provided for this module, you'll find some turret prefabs in the **Prefabs** folder. But feel free to create your own if you'd prefer.

As for bullets, you can design them yourself. A simple circle sprite (or anything else you like) will work just fine.

For now, you must have 3 different types of turrets:

- One with a low rate of fire and base damage of 0.3.
- One with a medium rate of fire and base damage of 0.2.
- One with a very fast rate of fire and base damage of 0.1.

Each turret must have a detection zone. When an enemy enters this zone, the turret should target and shoot it. It should always target the closest enemy.

Turrets shouldn't have too much range and must be placed near the road to detect enemies. If placed too far from the road, they shouldn't be able to detect them.

You must use a single script for all turrets.

When a bullet hits an enemy, it should be destroyed, and the enemy should take damage. The amount of damage depends on the turret's base damage.

Each enemy has 3 HP. So, depending on the turret, a bullet may deal 0.1, 0.2, or 0.3 damage.

When an enemy's HP drops to 0, it must be destroyed.



Since you'll reuse your work in the next module, don't forget to export your assets so you can easily import them into Module03.

Chapter VII

Submission and Peer Evaluation

Submit your assignment in your Git repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double-check the names of your folders and files to make sure everything is correct.



You should not upload the entire Unity project to Git, as this can unnecessarily increase the size of your repository.

- Make sure Unity saves as many files as possible in text format rather than binary. In Unity, go to Edit > Project Settings > Editor. Under Asset Serialization, set it to Force Text.
- Ensure that the .gitignore file automatically generated by Unity is present.



The evaluation will take place on the computer of the learner or group being evaluated.