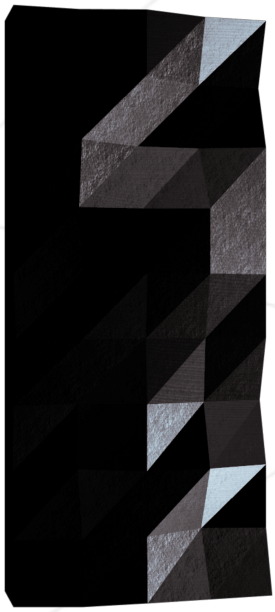

Unity Piscine - Module03

Advanced inputs and 2D GUI

Summary: In this document, you will find the Module03 subject of the Unity Piscine.

Version: 1.00

Contents

Chapter I

Instructions

- If you have problems installing the tools needed for your project on the 42 computers, you can use a virtual machine. In this case, you will have to :
  - install the virtual machine software on your computer.
  - install the operating system of your choice.
  - install the tools needed for your project.
  - Make sure you have the space on your session to install all of this.
  - You must have everything installed before the evaluation.
- Only this page will serve as reference. Do not trust rumors.
- The exercises have been ordered from easiest to most difficult. Under any circumstance you can submit or take into account an exercise if a previous one has failed.
- Be careful with the access rights of your files.
- You should follow the submit procedure for all you exercises.
- Your exercises will be corrected by your piscine peers.
- You cannot leave any extra file on your repository except the ones explicitly specify on you subject.
- Got a question? Ask your peer on the right. Otherwise, try your peer on the left.
- Everything you need can be found on the `man` or out there on Google.
- Read carefully the exercises: they may contain some features you should implement that are explicitly mentioned on the subject.
- Use your brain!!!

Chapter II

Foreword

As you saw earlier, you will be reusing what you created in Module02 for today's Module03. So you need to import your assets into your Module03 project.

Chapter III

Exercise 00: A simple menu

	Exercise :
Exercise 00: A simple menu	
Turn-in directory : `unityModule03`	
Required elements : `"Menu" scene and anything useful`	
Forbidden functions : `None`	

Create a simple scene Menu for our Tower Defense. Let's keep simple here. Juste make a Play, start or whatever button that will launch the game and a button that will close the application.

The Play button must lead to the ex01 scene, that doesn't not exist yet, but you will create in the next exercise.

Find a nice looking background an attractive layout !

Chapter IV

Exercise 01: Drag and drop

	Exercise :
Exercise 01: Drag and drop	
Turn-in directory : `unityModule03`	
Required elements : `"map01" scene, scenes and anything useful`	
Forbidden functions : `None`	

`http://docs.unity3d.com/Manual/Layers.html is your friend! It will help you manage the drag and drop as much as it will help you avoid that enemies/missiles/etc run under your turrets for instance.`

`Cooldown: In video game, the cooldown is the time an entity has to wait before it can do a certain action again. So for our turret is the time we have to wait before the turret can fire again.`

Create a bar at the bottom of the screen. It will feature the first 3 basic turrets the player can buy.

Create also a square around the road to accommodate the turret you will place.

To buy a turret, you have to drag and drop it from the bar to the place you want to place it.

Make sure that the elements of the `UI` stay in the foreground and don't go under other sprites !

The turrets is placed only if the targeted square is empty and the player has enough energy in their reserve.
If so, the cost of the turrets is taken from the player's energy reserve.

You must create this energy reserve.
It's up to you how this one works, does it fill up with time? according to the number of enemies killed? etc.

The bar must also display the damages, price and cooldown for each turret. You must also display in this bar the HP of your base and its energy.

You need a visual feedback. Change the turret color for example, to see at a glance if there is enough energy in our base reserve to bye a turret. You must not be able to move the turret if you don't have enough energy in reserve.

The color propriety is interesting because you can SET it as well as GET it. Hence, you can test the color of a sprite, which can happen to be very useful. Go read the doc for more information.

Chapter V

Exercise 02: Pause menu

	Exercise :
Exercise 02: Pause menu	
Turn-in directory : `unityModule03`	
Required elements : `scenes and anything relevant`	
Forbidden functions : `None`	

You will now add a pause menu when the player presses the Esc key. To manage the pause, you can look at the `Time.TimeScale`. this menu must propose resuming or quitting the game. If the player quits, a second menu appear for him to confirm his choice. If the player validate again, he returns to the main menu's scene.

Chapter VI

Exercise 03: I Am A Hero

	Exercise :
Exercise 03: I Am A Hero	
Turn-in directory : `unityModule03`	
Required elements : `"Score" scene, scenes and anything relevant`	
Forbidden functions : `None`	

You must create a screen that displays the score and the rank of the player at the end of the map (winning or losing). You must be make the enemy spawn stop when it has sent a certain number of enemies.
A score is calculated in the GameManager

You can create ranks from F to S and find creative titles! The only constraint is to give the player a rank that corresponds to the life and energy they have left with at least 3 different ranks.

The screen must feature a replay button is the player has lost or get to the next level if the player has wiped all the enemies wave.

You must also create a new map with an increased difficulty.

Create a classy end screen if a player beats your last level ! Always think of a nice way to end a game. It should never just go back to the title screen.

`If you want to change the difficulty, make sure you don't change the enemies' features on the original prefabs since this would modify their stats on EVERY map (prefabs are independent from the scenes). You must change the stats afterwards in a script or modify other game parameters.`

Chapter VII

Submission and peer-evaluation

Turn in 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 ensure they are correct.

You should not put all the files of a project on git, otherwise the disk space occupied by the repository will be unnecessarily increased. Here is how to configure Unity and GIT for an optimal use.

- Make sure that Unity saves as many files as possible in text form instead of binary.In Unity, go to Edit >Project Settings > Editor. Under textttAsset Serialization, you have to choose the Force Text Mode.
- check that the .gitignore file automatically generated by unity is present.

The evaluation process will happen on the computer of the evaluated group.