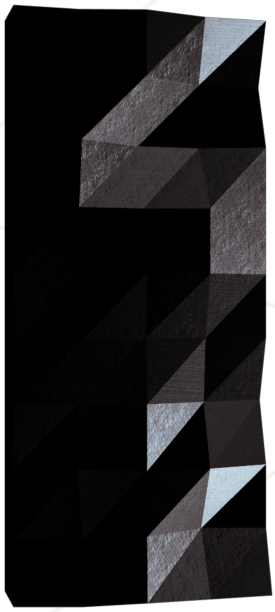

Unity Piscine - j03

Advanced inputs and 2D GUI

Staff staff@staff.42.fr

Summary: In this document, you will find the subject for the Day 03 of the Unity Piscine of 42.

Contents

Chapter I

Instructions

- 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.
- Think about discussing on the forum Piscine of your Intra!
- Use your brain!!!

At least one default constructor, a copy constructor, an

Chapter II

Foreword

Today, we're gonna play this game.

Chapter III

Exercise 00: A simple menu

	Exercise : 00
Exercise 00: A simple menu	
Turn-in directory : *ex*00/	
Files to turn in : `"ex00" scene and anything useful`	
Forbidden functions : `None`	

`If you've never played a Tower Defense, you should run the demo provided with the subject and play it for 5 minutes in order to get familiar with the genre's mechanics, since it's what you're about to be asked to create, basically.`

Create a scene with a menu for our Tower Defense. Let's keep simple, here. Just 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 exist yet, but you will create in the next exercise.

Find a nice looking background and an attractive layout!

`Today's program is heavy, so don't waste too much time trying to personalize your menu!`

Chapter IV

Exercise 01: Drag and drop

	Exercise : 01
Exercise 01: Drag and drop	
Turn-in directory : *ex*01/	
Files to turn in : `"ex01" scene 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 towers for instance.`

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

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

The bar must also display de damage, price, range and reload time of each tower as well as the player's life and energy. You must not write these values in hard but get them in the GameManager and the towers' different prefabs (provided in today's assets).

You need a visual feedback - change the tower color, for instance - to know in a wink if there's enough energy in the reserve to buy a tower. You must not be able to move the tower 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 : 02
Exercise 02: Pause menu	
Turn-in directory : $ex02/$	
Files to turn in : `Same scene as "ex01" and anything relevant`	
Forbidden functions : `None`	

You will now add a pause menu when the player presses the esc. key. There's a function ready in the GameManager to manage the pause. You just have to set true or false in parameter to stop or resume the game:

Protoype :

```
public void pause(bool paused);
```

This menu must propose resuming or quitting the game.

If the player quits, a second confirmation menu must appear. If the player validates again, they return on the main menu's scene.

Since you have to save yourself some time, the GameManager also features a function to speed up or slow down the game:

Protoype :

```
public void changeSpeed(float speed);
```

You're free to choose your implementation, but you need at least a button to speed up the game and another one to slow it down and pause it. You will save some precious time for your tests and evaluation. Pause and ChangeSpeed functions are compatible. You will automatically find the proper speed when you exit pause, even if you were playing

at fast speed.

A game with its own cursor is always classier. Find a way to change it! Go check the Unity documentation. You should find a solution in 5 minutes.

A cursor is provided with the resources today, but if you think it's ugly, find a nicer one!

Chapter VI

Exercise 03: Rox or sux?

	Exercise : 03
Exercise 03: Rox or sux?	
Turn-in directory : *ex*03/	
Files to turn in : `"ex03" scene 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.

A score is calculated in the GameManager so you can get it here or create your own.

You can create ranks from F to SSS+ or find creative titles! The only constraint is to give a rank to the player that matches the life and energy they have left with at least 5 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 waves.

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

`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

Exercise 04: Some more work?

	Exercise : 04
Exercise 04: Some more work?	
Turn-in directory : $ex04/$	
Files to turn in : `"ex04" scene and anything relevant`	
Forbidden functions : `None`	

`All the tower upgrades are already set in the prefab folder to help you save time.`

Place the towers is fun, but it's pretty basic. Let's upgrade them!

Create a radial menu popping around a tower when it's clicked.

The menu must propose an upgrade for the tower. You can upgrade the tower from level 1 to level 2, and from level 2 to level 3. The interface must show the cost of the upgrade to the player. Of course, the player can only upgrade a tower if they have enough energy to do so. When the tower is upgraded, the matching energy cost is taken off the reserve.

The radial interface also offers a downgrade to get the level 3 tower to level 2, and level 2 to level 1. The interface must show how much energy you can get when downgrading/selling a tower. You choose the cost but in general the selling cost is half the buying one.

A third button must allow the player to close the menu unless you decide to deal with it otherwise, right clicking an empty space for instance. You're free to implement it as you wish as long as the menu is shut flawlessly.

You must also create a slightly more difficult map to test the turrets and their de-

struction factor.

Chapter VIII

Exercise 05: For progamers

	Exercise : 05
Exercise 05: For progamers	
Turn-in directory : *ex*05/	
Files to turn in : `"ex05" scene and anything relevant`	
Forbidden functions : `None`	

One last thing is missing for the game to be completed: keyboard shortcuts! Rather than drag and drop towers one by one, it would be way easier to select them all with the keyboard.

Create a key mapping system to select the towers. For instance, pressing "q" or "1" replaces the mouse cursor with a turret. The player just has to click a free zone to place the turret.

The final touch: an emergency destruction spell if an enemy manages to pass your defense. Create a fireball in the bottom bar. You must be able to drag and drop it anywhere in the map to create an explosion. You must also create a keyboard shortcut for this fireball and a cooldown to stop the player from using it repetitively.

You will also find images that were not used in the previous exercises, like scope templates (pas sûr du terme en Anglais...). You can complete the interface and give a maximum of informations to the player so you can design a clear and demanding game increasing in difficulty.

Don't forget to 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.

`This time you will have no prefab or ready made scripts for the last features you should add. But if you made it here, it should not be a problem to you.`