16.08.2019

31263 / 32004 Game Development

Lab Week 6

Getting Started

1. Download the corresponding week's zip file from the Lab section of UTSOnline.
2. Unzip the project folder and open it in Unity. If there are any warnings about difference in versions, just continue. If this causes any red errors in the console once the project opens, notify the tutor.
3. Within the Weekly folders there are image and executable files starting with "Status…". These files give you a preview of what is expected for each point percentage below.
    a. If you are on Mac and the Status-100Percent App file won't run, hold control and click (right click) then select Open, if there is a security warning, acknowledge it an press open again.
    b. If you are running the Windows executable on the lab computers, you need to copy the entire executable folder into Windows(C:)/Users/<student number>/AppLockerExceptions. From there you can double run the .exe file.

Tasks

Points	Requirements
40%	• Open the project folder in Unity. Open Week3Scene if it is not already open. • Create an empty GameObject called "Managers". • Create a folder in the Project panel called "Scripts" • Create a script in the Scripts folder called TimeManager • For every 1 second that passes, TimeManager should print the time since start of game as an integer seconds value. (tip: use a private member variable such as "int lastTime" to store the time value that was last printed. Then use Time.time in Update() to check if a full second has passed) • Attach TimeManager to the Managers gameobject

50% (P)	• In TimeManager, use Input.GetKeyDown(KeyCode) to detect when the Spacebar is pressed, which should do the following:
	  ○ Pause and unpause (tip: see Time.timeScale)
	  ○ Print to the console "Spacebar pressed"
	  ○ Note: Update() is still called but Time.time is no longer updated

60% (P)	• Instead of Time.time, change the code to use Time.deltaTime and a private variable called “timer”. Keep track of time by adding deltaTime to the timer (e.g. timer += Time.deltaTime;) • Make sure that TimeManager starts printing from 0 (e.g. 0, 1, 2… etc). • On pressing the Enter (Return) key, the seconds counter should be reset to 0. ○ The reset should take place in a new private method called ResetTime() to make this functionality easily re-usable. It should be called when the game starts and then whenever the Return key is pressed.

70% (C)	• From the Prefab folder in the Project Window:
	  ○ Drag a RedPrefab into the Hierarchy Window, make sure the rotation is zeroed and set the position to (2,1,0)
	  ○ Drag a BluePrefab into the Hierarchy Window, make sure the rotation is zeroed and set the position to (-2,-1,0)
	• In TimeManager:
	  ○ Create a serialized private Transform array called transformArray. Select the Managers gameobject in the Hierarchy Window and change the size of the transformArray in the Inspector Window to 2. Drag the RedPrefab and BluePrefab instances from the Hierarchy Window into the array.
	  ○ Create a c# “constant” float member variable called “moveWait” and set it to 2.0f.
	• From the Start() method of the TimeManager
	  ○ Change the camera view from “perspective” to “orthographic” (tip: see Camera.main and Camera.orthographic).
	  ○ Set the orthographic size of the main camera to 2.0f.
	  ○ When you press play, what changes?

80% (D)	• In TimeManager, create a new private method called MoveObjects() ○ Once every <moveWait> seconds on the timer, move the colored objects in a synchronized square clockwise manner (e.g. when the timer prints out as 2 the RedPrefab should pop to (2,-1,0) and the BluePrefab to (-2,1,0), then move again at 4 seconds). • Challenge: Can you make it so that this movement can account for any starting positions of the two objects but assuming that there are only two objects to move and that they start with different x and y values? (not required, only for a fun and useful challenge)

90% (HD)	• Create another const float called “scaleWait” and set it to 4.0f.
	• Create the private method “ScaleObjects()” which should do the following to the transformArray elements:
	  ○ If an object’s local x scale value is greater than 1.5 then divide all scale values by 1.2.
	  ○ Otherwise, multiply all scale values by 1.2.
	  ○ Tip: see Transform.localScale
	• In ResetTime(), use InvokeRepeating(…) to call ScaleObjects() once every <scaleWait> seconds.

	• This behavior should obey the Pausing and Reset functionality (tip: see CancelInvoke(…))
100% (HD)	• When the "Escape" key is pressed, TimeManager should generate a random value between (0.25, 0.75) and start the following coroutine (tip: see StartCoroutine()). ○ "IEnumerator RotateObjects(float randomDelay){…}" • This coroutine should wait for <randomDelay> seconds, then rotate the two game objects by positive 90 degrees around the z axis, then repeat 3 more times (e.g. wait-rotate-wait-rotate and so on. The objects turn a full 360 degrees before exiting the method) (tip: see WaitForSeconds(…)) • This should obey the Pausing functionality but not the Reset functionality (i.e. the objects should finish their full rotation) • CLEAN, EFFICIENT, ELEGANT CODE!

Submission

When you complete the activity to the grade threshold that you want, you then need to:

1. Complete the "Status-StudentSubmission.txt" file in the highest level of the project folder.
2. Remove all other "Status-…" files and folders to reduce the size of your project.
3. Zip the entire project folder.
4. Re-name the zip file to **"[student ID]-LabWeek[week number].zip".**
5. Submit the zip file to UTSOnline for the associated link for this week in the Lab **before Monday 9am of the following week**.
6. Failure to follow any of these could result in a 0% mark for that week.
7. You will also **demo your submission to your tutor** at the **start of the following week's lab.**

If you finish the activity early, show it to your tutor before you submit it on UTSOnline so they can help you make some final corrections and mark it at that time.