Add the Timer and Scoring System

Hey everyone!

Super glad to see you all back here for game development using Unity! Today, we are going to focus on working with two new aspects of the game - scoring and timer! We will look at how to include a timer for level two and then we are going to look at adding a scoring system to our game! So, let's get started!

Let's open Unity hub and open the project that we have been working on.

Let's maximize the screen. In the hierarchy section, click on canvas, select level 2 and right click to choose UI and select panel option. Rename it as "level timer panel". Head to the colour section of the panel under the image window and change the alpha to 0 so that it doesn't look like a layer is present on top of it. Also, ensure to disable the ray cast target by unticking this box. We do this because we don't want this panel to be take any touch input. Now, jump back to the hierarchy window, right click on Level Timer Panel, select UI and click on text. Rename it as "timer text".

Now, I'm going to make sure that the timer text is positioned on the top left corner of the screen. To do that, we are going to select timer text and change the position and pivot by holding down the shift key and selecting the box at the first intersection on the top left corner. For selecting, press Shift + Alt key, then click the suitable option. Resize it as shown on the screen. Change the text to "Time Remaining". Also, tick the box that says best fit. Let the alignment to be on the left, but in the centre. We'll move the position of the text box to X as 50 and Y as -25.

Now, let's get started with the script. Go to the project window, assets folder and then inside the scripts. Right click, pick create and select C sharp script. Name it as "level timer". Now double click to open the code editor with Level Timer script.

Once opened, remove the start and update function as we don't need them. Begin by including UI function as

Using UnityEngine.UI;

And let’s define the reference of the text box created by writing code as

```
{
Public text levelTimerText;
Public int totalTime;
Public void StartTimer()
{
}
IEnumerator Timer()
{
Yield return new WaitForSeconds(2);
Int _counter = totalTime;
While (_counter > 0)
        {
        _counter --;
        levelTimerText.text = “Time Remaining: ” + _counter.ToString();
        yield return new WaitForSeconds(1);
        }
Debug.Log(“Game over show result screen”);
}
}
```

Inside the start timer function write the code as

```
StartCoroutine(Timer());
```

Now, jumping right back to unity editor. We are going to wait until it complies all the scripts. Go to level timer panel, click on to the inspector window, type level timer and press enter. Drag timer text to the Level timer text field and change the Total time to 5 seconds for now, which basically means that the level 2 has to run for 5 seconds only. This is done only for testing purpose.

Now that we have written the value, that doesn’t mean that it is going to start on its own, we are going to have to call the start timer function. To do that, get to game controller and then double click on level two gameplay to open the Level2Gameplay script.

As you can see that whenever the level two starts, we were calling the function "Init", so that all the animals are arranged in the grid. So, in the same function. We are going to write

//Start game timer

And then we are going to take reference of level timer script that we have written and then call the function start timer. Let's switch to Level2Gameplay and declare

Public LevelTimer levelTimer;

And to start the timer write

levelTimer.StartTimer();

Let's get back to unity editor and wait for the scripts to get compiled. Select game controller. Now we have to add level timer, before which I'm going to freeze it by clicking on this lock in the inspector window, this is just to ensure that it doesn't move. Then, I'm going to drag and drop level timer on to the box that says level timer. After which, I'm going to unfreeze the lock and run the project, to see what happens.

I'm going to now click the console window, come back to the game and select all the animals. Now that we are done with selecting all the animals, the "congratulations" screen pops up and we have cleared level 1. As you can see the timer for Level 2 starts after 2 seconds of screen switch only. Once the timer hits 0, we are shown the message "game over show result screen". Our level timer is done, but the screen that's going to have to pop up after that, we are yet to create.

Go to level 2 in hierarchy window. Right click on it, select UI and chose panel option. Now, change the colour of the panel and make it darker. I'm choosing to stick with the shown shade. Then right click on panel, choose UI and select image. Let's increase the size of the image that has been added here. Rename it to "level clear panel". We will put an image onto this. First, we will go to the inspector window to click on colour option. I'm picking orange. Increase the size of image and reposition it as shown.

Now, right click on the image again, select UI and click on text. Align the text box to the top center as shown. Expand it so that it is stretched from one end to the other, increase the size to whichever amount you want, align it to the centre and tick the little box that says best fit. Write “well done” in the text box. Increase the size to 128 and bold it. Change the placement of the text by pushing it a bit down, entering a value of -25 into the Y box in the Rect Transform window. Rename it to title text.

Select image, right click, select UI and then choose text. Increase the size of this text box as well and align it to the centre. Change the text to Score:, make the alignment to center, font size to 81 and ensure to choose the best fit option again. Rename it to “Score text”. Now, right click on the image, select UI and chose button. Reposition it as shown and increase the size of the button. Expand the button, click on the text, choose best fit and then type in “Play again” in the text field. Rename the button to “Play again button”. Change its colour to green. As we have already discussed in previous videos that you can create a prefab out of level create panel. To do that, go to the asset folder, select prefabs and drag level clear panel and drop it into the prefab. Delete the “level clear panel” present in the hierarchy window.

Now, let’s instantiate the level clear panel by writing the code in the LevelTimer script by defining it as

Public GameObject levelClearPanel;

Disable the debug log by using // and writing

GameObject _lvlclear = Instantiate(levelClearPanel);

We’ll declare one more function as

Public Transform parentCanvas;

And make the game object by adding

, parentCanvas

Delete the debug log

Go back to unity editor and let it compile. As you can see, it is asking for level clear panel’s reference. So, go to prefab folder, drag and drop level clear prefab to level clear panel field. Next, drag canvas from the hierarchy window and drop it to Parent canvas field. Let’s run the game and see what happens.

As we can see the game is working fine so far. But we haven't written any functionality on the play again panel as of now. Basically, there are two functionalities that we have to write in here, one is play again. Play again is going to restart the whole scene again or the whole game again, and score is going to display the score which we will do once we implement the scoring system. So, let's implement the functionality of play again button.

To do that, go to the asset folder in your project and double click on level clear panel. If you remember, we had already written this functionality for level 1 time over. Here, we had the button called replay which was starting the whole game again. We are going to use the same functionality in play again button as well. So, go back to the asset folder and double click on Level Clear panel. Click on add Component in the inspector window, type in “replay game” and press enter. Now, go to play again button, go down in the button section. Click on + to add on click event. Drag and drop level clear panel in the object field and from the drop down, select replay game and click on play again function. Click on the small arrow in the hierarchy window to go back to the main window.

So, now that we have implemented the timer, so, we will implement the scoring system. To do that, go to hierarchy window. Just like we have created level timer panel, we can also directly right click on it, duplicate it and rename this to “score panel”. Expand it and rename the “timer text” to “score text”. We want it to be aligned to the right side, so, select score text, hold on to shift and change the position and pivot to right top side. Change the value to -50 for X and – 25 for Y. Change the alignment to right and change the text from “time remaining” to “score”. Next, to add and update the score, we have to go to project folder, assets, scripts, right click on it and create a new script called “GameScore”. Go to score panel, remove the level timer script that is attached to the score panel, because this is not a level timer. So, select “games score” in the scripts folder and drag and drop it on the score panel. This is another way to attach scripts. Now, double click on score panel to open the code editor.

Remove the update function as we don’t need it. Let’s begin coding by defining a function

Public UnityEngine.UI;

And declare the variables as

Public text scoreText;

Int score;

As soon as the scene is started, we would like to set the score by writing

```
{
Score = 0;
scoreText.text = "Score:" + score.toString();
}
```

Whenever the score is updated or reduced, we will call the function

```
Public void UpdateScore(bool _increaseScore)
{
        If (_increaseScrore)
        {
                score = score + 2;
        }
        Else
        {
                score = score – 1;
        }
        If (score < 0)
        Score = 0;
        ScoreText.text = "Score: " + score.ToString();
        }
}
```

Go back to Unity editor, select score panel. Drag Score text from hierarchy window to Score text field. To increase or decrease the score, select level 2 and double click on LineDrawing script to open the script. This is the script where all logic is happening with respect to start or end animal or the line is drawn or not.

Let's create 2 variables after parentcanvas by writing

Defines.AnimalsList startAnimal;

Defines.AnimalsList endAnimal;

Next go to start line drawing function and define

startAnimal = AnimalScript.startAnimal;

Change the defines to game object by rewriting it as

GameObject startAnimal;

GameObject endAnimal;

Next, go to end line drawing function and write after the if loop

endAnimal = AnimalScript.endAnimal;

So, if you see, there's one condition that we have been working on if animal script is not equals to null, then destroy the line drawing. So, let me just go to unity and run the project and see how the lines are getting drawn. I just want to test whether the lines are getting created or not. As you can see that the lines are not getting created properly. So, click on play button again and switch to code editor to the linedrawing script.

In the end line drawing function you see that there is a small bug that we left initially. That is, we have said that if the animal script dot end animal is not equal to null, then destroy current line object, but rather we should say if the endpoint is not on any of the animals, then we should destroy this line. Otherwise, we should continue using this line. So, change the exclamatory mark to equal to. Also, add an else condition by writing

```
Else
{
endAnimal = AnimalScript.endAnimal;
}
```

Here, we would like to create a function which decides whether we are selecting the correct animal and dropping it on the right animal. For example, if I draw a line from lion to leopard, then it will be wrong answer. But if I draw a line from a lion to probably a giraffe, or a rhinoceros, then it will be correct answer because a lion can eat a giraffe and rhinoceros.

So, write a function as

```
Bool UpdateScore()
{
If(AnimalScript.endAnimal == null)
Return false;
Defines.AnimalsList                    _startAnimal                    =
startAnimal.GetComponent<AnimalType>().animalType;
Defines.AnimalsList                    _endAnimal                      =
endAnimal.GetComponent<AnimalType>().animalType;
If(-_startAnimal == Defines.AnimalsList.Lion)
        {
        If(_endAnimal   ==   Defines.AnimalsList.Giraffe   ||   _endAnimal   ==
        Defines.AnimalsList.Rhino)
                {
                Return true;
                }
                Else
                {
                Return false;
                }
        }
Else if(_startAnimal == Defines.AnimalsList.Sparrow)
        {
        If(_endAnimal == Defines.AnimalsList.Grasshopper)
                {
                Return true;
                }
        Else
                {
                Return false;
                }
        }
Else if (_startAnimal == Defines.AnimalsList.Snake)
        {
        If(_endAnimal   ==   Defines.AnimalsList.Sparrow   ||   _endAnimal   ==
        Defines.AnimalsList.Fox)
```

```
                {
                Return true;
                }
                Else
                {
                Return false;
                }
        }
Else if (_startAnimal == Defines.AnimalsList.Fox)
        {
        If(_endAnimal == Defines.AnimalsList.Rat)
                {
                Return true;
                }
        Else
                {
                Return false;
                }
        }
Else if (_startAnimal == Defines.AnimalsList.Eagle)
        {
        If(_endAnimal   ==   Defines.AnimalsList.Snake   ||   _endAnimal   ==
        Defines.AnimalsList.Leopard)
                {
                Return true;
                }
        Else
                {
                Return false;
                }
        }
Else if (_startAnimal == Defines.AnimalsList.Leopard)
        {
        If(_endAnimal == Defines.AnimalsList.Deer)
                {
```

```
                Return true;
                }
        Else
                {
                Return false;
                }
        }
        Return false;
        }
}
```

You can keep on modifying the combinations in food chain as per your requirement and complexity that you would like to bring in to the game.

Now, go to end line drawing and when there is an end animal available what we want to do is to update the score. For that, we have written the function in games score scripts. So, go to the parent canvas function line and write

Public GameScore gameScore;

And in the end line drawing function, write the code for it. Before that, delete the if condition in the update score function as shown on the screen as it is not needed now. Come back to else condition and write

Bool _increaseScore = UpdateScore();

gameScore.UpdateScore(_increaseScore);

One more thing that we can add to this functionality is whenever there is a correct answer, we would like to make the line that we are drawing between both the animals to be green. And whenever it's a wrong answer then the line colour to show red. To do that, continue writing the code as

If(_increaseScore)

currentLineRenderer.material.color = Color.green;

else

currentLineRenderer.material.color = Color.red;

Now, let us test all the code that we have written! We have written good amount of code so far, so it is important to check if everything is working fine as we intend for it to! Let's go back to unity editor and let it compile. Before running the project, let us go to level 2 in hierarchy window, drag score panel to the game score field present in the line drawing script. Now that we are done with all this, let us run and see what happens.

As the level 2 gameplay time is too short, let's increase by selecting Level Timer Panel and change the total time from 5 to 60 seconds. Let's launch and see what happens now.

For testing purposes, we are going to connect deer to lion, to see if it's correct or wrong. We can see the line turning red because it is the wrong answer. Now, let's connect the lion to rhinoceros, implying that the lion can eat the rhinoceros. It can be seen that the line has turned green. Now, if I imply that a leopard can eat a deer, it has given me a null pointer exception. Let us see what a null pointer exception turns out to be.

Now, go to project, assets, prefabs and double click on level clear panel. In the level clear panel, we have to display the score. For that, go to assets, scripts, right click in the window, select create and chose C-sharp script. Rename it as LevelClearScript. Select LevelClearPanel and click on Add Component, type Level Clear Script and press enter. Double click on it to open the level clear script.

Delete the update & start function for now and include the component by writing

```
Using UnityEngine.UI;
```

The purpose of the script is to update the score on the final screen. So, let's begin by writing

```
{
Public Text scoreText;
Public void UpdateScore(int _score)
        {
        scoreText.text = "Score: " + _score.ToString();
        }
}
```

So, this is the only functionality that we need. For the other functionality like the button functionality we have already written the code. Let unity compile this code. Click on the back arrow in hierarchy window to go to the scenes list.

To update the overall score, we will do so from the level timer panel. Double click on Level Timer to open the script. Let's add a code by writing

_lvlClear.GetComponent<levelClearScript>().UpdateScore();

Define the public class as

Public GameScore gameScore;

Switch to GameScore Script and write the function outside start function as

```
Public int GetGameScore()
        {
        Return score;
        }
```

Switch back again to the Level Timer script and inside update score write

gameScore.GetGameScore()

This code will return the current score of the game and we are done with this script.

Switch back to unity editor, let's freeze level timer panel, drag and drop score panel from hierarchy window to the game score field. Click on unlock to unfreeze it. We have now completed all the functionalities that we wanted, so, let's run the project and see how everything is working together.

We are able to select all the animals as usual and after that is done, let's connect animals in the second level. Let's draw a line from lion to rhinoceros and then from lion to giraffe. We can say that a fox can eat a rat and that a snake can eat a sparrow. We are able to create a food web and every time we get something right as in every time my line turns green, it can be seen that 2 points gets added to my score. For instance, if we select eagle and connect it to snake, implying that the eagle eats snake, my score has increased by 2 points that is from 12 to 14. Similarly, if I draw a line from eagle to giraffe, which is definitely not the right answer, you can see that the line has turned red and that my score has been deducted by 1 point. Once the game is finished, we can see that the score is still not visible.

Let’s open the console to check why this is happening. The console is showing that a mistake has been committed and we know exactly how to fix this. To do that, go to project, assets, prefabs and double click on level clear panel. As it can be seen that we have set the value of text box but the reference is not taken. So, drag score text and drop it on to score text field to reference it. So now, we have fixed the bug and our pop-up screen should work.

Let’s get back to our scene. Select level timer panel and change the timer to 10 seconds for now, just to test the functionality. One more thing we have noticed earlier is that our pop-up screen which was orange also had lines appearing on top or in the front of it.

So, what's happening here is we are just reducing the Z value of the panel. To make it work, switch to level clear script and write a start function as

To do that, go to unity editor and select level clear panel and change the z value to -0.02, so that it always comes on top.

Let’s launch the project and see what happens. I’m going to connect the animals and as soon as our timer is up, we can actually see the score come up in our pop-up screen! With regard to the text there, we have no errors! But the lines still continue to come in front of the screen.

So, we’ll go to LineDrawing script and delete all the lines that we have created so that it doesn't interfere with the panel. Before doing that, let’s create a list by writing

List<GameObject> lineOjjectsList = new List<GameObject>();

Let’s scroll down to end line drawing function and write the code after line renderer as

lineObjectsList.Add(currentLineObject);

and write a new function after end line drawing as

public void DestroyLines()
{

```
While(lineObjectsList.Count > 0)
        {
                GameObject _tempObject = lineObjectsList(0);
                lineObjectsList.RemoveAt(0);
                Destroy(-tempObject);
        }
}
```

We should call this function in level timer script before we show the score panel. To do that, we will define the function

Public LineDrawing lines;

And inside the timer enumerator, let's write

Lines.DestroyLines();

These lines of code will help us hide the lines and the show the panel.

Let's go back to unity editor and assign the reference to it. To do that, select level timer panel and drag Level 2 to drop it in Lines field. Let's test our whole game to check if all the functionalities are working as expected.

Let's get through with the first level like we always do! Now, on to the second level, connect leopard to the deer, fox to the rat. Our timer is up, however both lines were correct, so the score is 2, which has been updated and the best news is that the lines don't interfere with the pop-up screen anymore! If we click the play again button, it is restarting the whole game as seen!

With that, we have finished the score and level timer. One last thing is to select level timer panel in the hierarchy window and set the duration of the timer to 60 seconds, it was set to 10 seconds earlier as shown.

So, we have come to an end of the video where we have completed the timer and scoring system! In the next video, we will learn to create the welcome screen and publish the game!

I must say, great job for having come this far! It has been an amazing journey, I'm very excited to see you for the last leg of it! Worry not, I will see you very soon, until then, do keep practicing and excelling in being the super game developer that you are!