Creating Animal Food Web

Hey everyone! Welcome back to game development tutorial using UNITY!

Today, we are going to create a food web of all the animals that we had selected in level 1. So, let's go ahead and open Unity hub. I'm going to select the food web project. Let's maximize the screen now.

So, if you remember in our last video, we had created a prefab called animal image which we had used to create a grid. So, if we go to hierarchy then canvas and open level 2, all the animals are added in to the grid.

If we go to game controller, we will see that in level 2, we had taken animal image as one of the prefabs. Now, we will select animal image from the project window and click on 'Add component'. Type 'Animal Type' and press enter to assign animal type of component. This is done to identify which image corresponds to which animal.

To add animal type image, select game controller from hierarchy window and double click on Level 2 game play script option in inspector window. It will open the visual studio with the script of level 2 gameplay. Now, we need to assign the animal type to each and every image that we are creating. So, let's write a code for the same as shown on the screen

_tempAnimal.GetComponent<AnimalType>().animalType = _animal;

What it does it that the temp animal image that we have created get accessed by component and we assign the animal that we are using to generate the image here. So, now the animal type is assigned to every animal.

Next, go back to unity editor and click on play. As you can see, the grid of all animals is created. Let's select Level 2 from hierarchy window and expand animal grid. You can see that clone image of all animals are created and it is showing under animal type.

Press the play button again to stop. Go to the asset window, right click and select create. Click on C sharp script and rename it to 'AnimalScript'. Double click on it to open the visual studio. We have created this script to detect certain events like selection of animal through mouse click that happens on the animal images.

Along with that, we're going to use this script to create a logic around selecting first animal, drawing a line from one animal to another animal.

So, let's get started writing a code as shown on the screen. We'll write

Using UnityEngine.EventSystems;

Extend the public class with

, IPointerDownHandler, IPointerUpHandler, IPointerEnterHandler, IPointerExitHandler

So, I've extended all these interfaces. Next, we'll quickly delete the start & update function.

As of now, the interfaces are showing an error because I have to extend and write their function. To do that the easy way, right click on the DownHandler and select Quick Fix and the first option in it. You'll see that it automatically creates the public function for the particular interface. Delete the internal function as it is not required.

Similarly, we can define the function for UpHandler as

Public void OnPointerUp(PointerEventData eventData)

{

}

Similarly, write the function for enter & exit handler too. You can do that through quick fix or you can choose to write it manually.

So, we have finished writing all the functions for interfaces of pointer down, enter, exit and up. Now, we will define 2 public variables as

Public static GameObject startAnimal;

Public static GameObject endAnimal;

We have made them static as we want to use them from any other script as well as we want them to contain only one value at the time.

In the pointerdown function, we will define the condition as

{

startAnimal = gameObject;

}

And in the pointerup function, we will define it as

{

startAnimal = null;

}

We'll go back again to pointerdown function and write

endAnimal = null;

as we do not want to keep the traces of other animals once overwritten

Also, whenever a pointer enters into the image of any of the animal, I want to write the condition in the pointerenter function as

If(eventData.pointerEnter != startAnimal)

{

endAnimal = gameObject;

}

What does this function do? So, whenever you point or click on any image, the function gives the data associated to that particular image. In our case, we want to detect the particular animals whenever we are on it or click on it then that animal should not be equal to the start animal, which is the animal on which we have already clicked. Don't worry about it for now as I'll explain it while playing the game.

For pointerexit, once the animal is selected, I want to end it and exit that particular animals. So, I will write the code as

endAnimal = null;

So, let's just quickly understand it again in unity editor. Go to Assets, prefabs and select Animal image. Click on Add component in the inspector window and type Animal Script. Press enter to add it. Run the game and select the animals in

level 1 as usual. You will notice the changes happening in the hierarchy window automatically.

Once the grid page loads, select the event system and maximize the folder in the inspector window by dragging it upwards from the bottom & click on lock for a moment. As you can see in the log that the position value is constantly changing depending on the movement of pointer on the screen.

So, to click & select the animals, OnPointerDown function gets called where it selects it as a game object. Once the initial animal is selected and then the mouse pointer clicks on any other animal image, then OnPointerEnter function is called. OnPointerExit gets called once you are finished selecting the end animal. Lastly, OnPointerUp is called whenever the left mouse button is pressed and it is released post selection of initial animal.

Now, go back to unity hub. In the asset folder, select scripts. Right click, select create and C-sharp to create a new script. Rename it as 'LineDrawing'. Line drawing basically ensures that when we click on any animal, drag it from there and release that on any of the other animals, then a line is drawn between the first and the last animal.

So, double click on line drawing script to open the code editor. To begin with, we will define some variables. We will start by writing

Vector3 startPosition;

GameObject currentLineObject;

LineRenderer currentLineRenderer;

Each line renderer requires material for which we will write it as

Public Material lineMaterial;

Next, define the thickness of the line by writing

Public float lineThickness;

Also, define the canvas which will ensure to draw the line on a particular canvas. Write the code as

Public Canvas parentCanvas;

Let’s go to the start function and begin writing the code for parent canvas as

parentCanvas = GameObject.FindObjectOfType<Canvas>().GetComponent<Canvas>();

What this code does is, it finds the object of canvas type and then we will take the canvas component of it.

Next, we will go to the update function and write the code for mouse press functions as

If(Input.GetMouseButtonDown(0))

{

}

This first condition will tell us if the mouse button is pressed or not. Next, we will write the other condition

Else if (Input.GetMouseButton(0))

{

}

Else if (Input.GetMouseButtonUp(0))

{

}

The second condition will tell us if the mouse button is still pressed and it is moved on the screen and the third condition will tell us as soon as we release the click on mouse button.

Now, we want to start drawing a line as soon as the mouse button is pressed down. To do that we will define the function in public class as

Void StartLineDrawing()

{

}

Another function would be to preview the line as we are dragging it anywhere on the screen. To do that, we will write

Void PreviewLine()

{

}

And the last function would be to stop drawing the line as soon as the mouse button is release. So, let's write it as

Void EndLineDrawing()

{

}

So far, we are able to get the coordinates from mouse function in general coordinate system but it is not understood by canvas coordinate system. Hence, we would require to convert the positions received from mouse pointer to canvas coordinates. So, we will write one more function for it above the line drawing function as

```
Vector3 GetMousePosition()

{

Vector 2 movePos;

RectTransformUtility.ScreenPointToLocalPointInRectangle(parentCanvas.transform as RectTransform, Input.mousePosition, parentCanvas.worldCamera, out movePos);

Vector3 positionToReturn = parentCanvas.transform.TransformPoint(movePos);

positionToReturn.z = parentCanvas.transform.position.z – 0.01f;

return positionToReturn;

}
```

Now, let's write the code for the start line drawing function as

startPosition = GetMousePosition();

currentLineObject = new GameObject();

currentLineObject.transform.position = startPosition;

currentLineRenderer = currentLineObject.AddComponet<LineRenderer>();

currentLineRenderer.material = lineMaterial;

currentLineRenderer.startWidth = lineThickness;

currentLineRenderer.endWidth = lineThickness;

So, this code gives me the start position and set it as a marking point and current line renderer will have the line material with a particular thickness at both the starting and ending point.

Let's write the code for preview line function which will show the line drawing from the starting to ending point as

currentLineRenderer.SetPositions(new Vector3[] { startPosition, GetMousePosition() });

As soon as we release the mouse button press, we need to reset it to the start position. Let's start writing the code for End Line drawing function as

startPosition = Vector3.zero;

currentLineObject = null;

currentLineRenderer = null;

Next, let's call these functions in the update function as

StartLineDrawing();

in first condition

PreviewLine();

in second condition

EndLineDrawing();

In third condition respectively.

So, let's go back to unity hub and let it compile all the scripts we have just written. Select Level 2 from hierarchy and unlock the inspector window. Drag the level 2 preview log window to the bottom. Now, click on Add component and type 'Line Drawing' and enter. Line drawing script gets added.

Now click on asset and create a new folder called Material. Open the material folder and click on + button. Select material option and rename it Line Material. In the inspector window, and select shader option. From the drop down, select UI and default option. Click on tint and select the black colour and close the window.

Select Level 2 again from hierarchy window and drag line material from asset section to Line material field as shown. Change the line thickness to 0.05. Click on play to test the progress made so far.

So, as you can see that our lines are getting created however, there is no fixed state. We need to ensure that the lines start from one animal and end with another. We will get to fixing that. But for now, we can draw lines! Let's switch back to the edit mode and go to the script.

As we remember to create 2 static variables i.e. start and end animal in the animal script. So, go to the line drawing script and in the mouse pointer down function, we will define

If(AnimalScript.startAnimal != null)

It means that we would like to start drawing the line if the start animal is selected and not from any blank area.

In the second condition, we will define

If(AnimalScript.startAnimal != null)

It means that we would like to preview the line only if the start animal is selected.

In the endline drawing function, we will write the code as

If(AnimalScipt.endAnimal != null)

{

Destroy(currentLineObject);

}

It means that line should destroy itself if the end line is not present on the end animal. For example, if the deer is the start animal and line is dragged to empty space, then the line should get destroyed.

Let's go back to the unity hub and test our code. Click on the play button and maximize the game window. Select all the animals from the screen to go to the next level.

On the animal grid, start drawing the line from leopard to deer. If you properly draw a line from one image to another, you will be able to draw it just like you saw on the screen.

With that, this video comes to an end! As for the next video, we will work with creating a scoring system for our game! This means that if and when I draw lines connecting the right animals in the right order to create a food web, I will get points, however if I go wrong by connecting incorrect animals to each other, my points will reduce accordingly! This is what we are going to look at in our next video and I hope you're as excited for it as I'm!

Hope to see you very soon! Bye Bye!