Animating and Selecting Animals

Hello friends!

Welcome back to game development using UNITY!

So, in the last video, we had created our scene by placing animals into the theme. Now, we are going to look at how to animate those animals so that they can move and we can make the scene more interesting. Also, we are going to write the logical input to select different animals so that they can be used in the food web.

So, let's get started! Open your UNITY hub and select the project titled 'food web'.

Now, let's maximize the screen. We are looking to animate our animals, so, let us start by adding animation to our rat. We want this rat to move from the current position to behind the bush position, where he will stay for around five seconds and then move back to the original position. To do that, we must select rat from the hierarchy window, go to the window option in the top bar and select the "animation" option. The animation window is open. Create new animation. Before doing that, do not forget to go to the assets animation folder, create a new folder called "rat" and title your animation as "rat_walk".

In the animation window, start recording the animation in its 1st position. For the 2nd position, I want the rat to move from here to there in 2 seconds. So, I will just set the marker to 2 second indication and then move the rat around from here to this position. Now, I want my rat to remain behind the bushes for at least 5 seconds, so I'm going to add the previous 2 seconds to 5 seconds to make it 7 seconds in total. Now, I will set the marker to 7 seconds. As I want the rat to stay here for a while, hence, I'm just minimally changing its position, from a point 9 to point 8 so that it's position can be reset. I'm going to stop the recording and see how the animation plays out. I can see the rat animation is working out well.

Next, we are going to move the grasshopper from one side of the screen to the other. Select the grasshopper from the hierarchy window and position the grasshopper to the extreme left, outside of the screen. Repeat the same steps for adding animation just like we did for rat. Create a new folder called "grasshopper" in the Animations folder and name the animation as 'Fly'.

Start the recording. The initial position of grasshopper is (-1083, -375). Keep a note of it for resetting the object. I want grasshopper to move from the current position to this extreme right, outside the screen position. Zoom out the animation timeline as I want the animation to run for around 10 seconds, so that it moves slowly. Simply drag and move this grasshopper from start to end position. Now I want to reset the position. To do that, slightly move the line on the frame again and set (-1083, -375) in position field. Now, stop the recording and play the animation. As you can see, the grasshopper is flying from this one end to the another end and then reappearing again at the starting point.

Next, we are going to work on animating the deer! We are going to do two things as far as the deer is concerned, we ensure that the deer has a walk cycle and then we need to move this deer from one side of the screen to another side of the screen. Let's select deer from the hierarchy window and disable bushes for a while to see how the deer walk will happen.

As of now, the deer is just standing with no animation attached to it. First go to assets, images and animals. From the last time, you should remember that we broke down the single image of the deer into multiple images in the previous videos. Expand the arrow mark to see the walk cycle animation for the deer. Proceed further by creating a new folder. Go to projects > assets > animations > create a new folder called "Deer". In the animation, simply right 'Walk' and save it. So, now a new animation has been created for the deer. Now select all the images of Deer animation and drag on to the animation panel. Animation for the deer is done. Play and see how it works. As we can see the speed is fast. Reduce the speed by dragging the small button to 1-minute and 30 seconds mark. Play and see the animation again. Now, the walking speed of deer is fine and we are done with deer animation. Click on the 'sprite editor' after selecting the deer asset to see the frames used in animating the deer.

But I want the deer to come out and walk all the way through and reset back to its original position.

Let's explore a way to do this through code! So basically, my dear should start from slight left of the center part of screen. Let's make the bushes visible. It should walk all the way to the end of screen and reappear again to its original position. The default position is (-284, 200). This is essentially a loop and to make that happen, go to asset folder, create a new folder called scripts. In the script folder, right click, select create and click on C sharp script. Name it as “DeerWalking”. Double click on it to open the Visual studio for writing a program.

So, there are two functions that have already been given to us - one is the start function which calls the object to which this script is attached and the update function which gets called continuously with each frame of the animation. First we need to define two fields. In the next line of open parenthesis, write

public Transform startingPoint;

This is going to be the starting point of deer, that is, the left side from where it starts moving. Define another variable called

public Transform endpoint;

This defines the end point of deer. Let's define one more variable called speed by writing

public float speed;

This is the speed with which the deer will move from start point to end point. As we are not initializing any object so let’s delete void Start() function as shown.

In our void Update() function, we are going to create a simple variable called

{

float step = speed X Time.deltaTime;

In this step, what happens is that the deer will take the number of steps based on the speed and time. And the delta time is the difference between two frames that are being drawn between 2 image references. As we are attaching this script to the deer, we are going to take the transform position of the deer. So, we will write

transform.position = Vector3.MoveTowards(transform.position, endpoint.position, step);

So, this takes the current position, the end position and how much distance deer want to travel. Now once it reaches the end point, we want to reset the position to the starting point. So, let's check the condition by writing a code

if(Vector3.Distance(transform.position, endpoint.position) < 0.01f);

What we are checking here is if the deer has reached almost or at the end point, we should reset the position of deer to start point. So, we will write

{

Transform.position = startingPoint;

}

This is going to continuously keep the deer moving from one position to another. Now let's set the values of all the public variables that we have defined.

To do that, go back to Unity editor and save the scene to update the script for DeerWalking.

Now, select Deer from Hierarchy window and click on add component from Animator window. Select the name of the script that we have created by typing DeerWalking. Once entered, it is showing me three public variables i.e. the starting point, End point and speed.

As of now, I have not created any starting point. Right click on Canvas in Hierarchy window and select create an empty object and move it near to deer object. Rename it Deer Starting Point. Right click on it and select duplicate. Drag it near Deer Starting Point and rename it Deer End Point.

To set the starting point, select Deer Starting Point and drag the box to (-1080, -200) position and to set the end point of deer, select Deer End Point and drag the box to (-1244, -200) position. Use the move tool to avoid misbalance on Y-axis.

But I want my deer to be present behind the tree, so I'm going to select Deer, Deer starting point & Deer end point and place them above Tree object. Now, it will look like the deer is behind the tree and bushes.

Select Deer option and under Deer Walking script, select Starting Point. A new window opens. Double click on Deer Starting Point option. Similarly, select Deer

End Point for End Point option. Set the speed as 5. The speed is fast. Let’s change it to 2 now which is optimum.

Now, let’s see how the whole scene is working. Click on Play button to launch the scene. We can see our scene is working perfectly.

Next, what I want in my game is that whenever I select an animal, that animal should get selected and disappear from the whole scene and get added to a list from where we can access it while creating the food web.

To do that, minimize the canvas first. Click on ‘+’ and select create empty object and reset the object by clicking on the 3 dots in Transform section and select Reset. Name it as “GameController”. In the same Scripts folder, create a new script called “GameController”. Next, select the game controller object from hierarchy window and click on Add Component. Type ‘GameController’ and double click to open the visual studio. Let's delete the start & update function and understand what are we going to put in game.

So, game controller is going to be the script that will handle all the states of your game. By states I mean, when the game starts there will be a welcome screen. Next, a second screen called level scene that comes before the scene of gameplay level, third screen with our food web scene or level one completion screen. So once that is complete, then you go to the level two screen where you create the food web and the last, game over screen. Let’s define these states now.

To do that, we can start by defining an enumeration. This game’s enumeration is going to contain all the game states. You can initialize them like

{

Enum GameStates

{
WelcomeScreen,
Level1_Gameplay,
Level1_GameOverScreen,
Level2_StartScreen,
Level2_GamePlay,
Level2_Gameover,

GameComplete
}

We can add more states to this as we go along, but these are some of the major states in which we are going to create the game. So, for now, let's not worry too much about the welcome screen because that's something that we are going to cover in another video.

Before proceeding further, let’s define these states in a different script all together to avoid any confusion. To do that, go back to Unity editor. Go to script folder in your assets, right click and create a new C sharp script and name it as ‘Defines’. Double click on it to open the Visual studio. Delete the existing functions.

Delete the MonoBehaviour and change the public class Defines to

public Static class Defines

and make it static. Now, go to GameController tab and copy the whole code and paste it under defines.

Now to access the states from outside, the game states class is to made public by changing it to

public enum GameStates

Come back to the Game Controller tab and begin by creating a function

Public void UpdateGameState(Defines.GameStates _gameState)

Let's define a global variable as

{
Defines.GameStates currentGameState;

Let’s focus on Level 1 Gameplay screen for now. So, we are going to define a current status by writing

{
currentGameState = _gameState;
}

Update Game state is the function that we are going to call whenever we want to change the state of the whole game. Once this is called, the statement saves the current game state. Next, let's write a switch function as

```
switch(_gameState)
{
case Defines.GameStates.WelcomeScreen:
{
}
break;
case Defines.GameStates.Level1_Gameplay:
{
}
break;
```

This is all about game controller. We are going to start putting code into this file for all states. Let's go back to unity editor and see what we would like to achieve first in the Level 1. The first thing that we want is to be able to select all the animals by clicking on them.

Hence, we will create another object and name it as Level1GamePlay. Accordingly, create a C sharp script in the Asset folder and name it as Level1GamePlay. Double click to open the new tab for it. Again delete the start & update function.

In the level 1 gameplay, we want the users to select the animals and once they get selected, they should disappear.

To do that, we are going to create a function for animal selection by taking game object for it and assign a unique ID to each of them so that they can be identified once they are selected. After selection they will disappear from the screen. For that, write the code as

```
{
Public void SelectAnimal(GameObject _animalObject)
{
Destroy(_animalObject);
```

}

Go back to the unity editor and save to refresh it to update the script. Now, select Level 1 gameplay from hierarchy window and reset to initial position. Click on Add component type ‘Level1Gameplay’. Maximize the Canvas and go to the animals one by one.

Let's start with fox from the top. Right now, it is just an image as we haven't added any event or function to it. Click on Add component and type ‘Event Trigger’. Now click on ‘Add New Event Type’ to open the options and select ‘PointerClick’. Click on ‘+’ to add from list.

Now it's asking for the function that you want to call. So to add that function, drag Level1GamePlay from hierarchy window and drop it to the pointer click variable. From the dropdown menu, select Level1GamePlay and select the function that we have created which is SelectAnimal(GameObject). Now, you will be asked for the parameter or AnimalD. _animalObject is the parameter in our case which in other terms are the animals. First parameter we can define is Fox. Drag Fox to the Game object field. So, whenever I click on the fox, it gets disappeared.

Similarly, we'll do it for all the other the animals. Follow the same steps for adding Event Trigger > PointerClick > Drag Level1GamePlay and drop it to the pointer click variable > Select Level1GamePlay > Function as SelectAnimal(GameObject) > Drag Giraffe to game object. Repeat the steps for all animals.

Great. So what happened here is that whenever any click event happens on the animal, it is going to select the particular animal function with parameter as that animal in Level1GamePlay script. Let's run and see what happens.

I am clicking on the leopard, but it is not getting selected. It is because there is a Raycast that is blocking the click function. Let’s select the Forest Background and you can see that Raycast is enabled which means that the Forest Background image is also interactive at the moment. But in our game we don’t want all the objects to be interactive except animals. Hence, let’s disable it by unticking that. Let's disable for all tree, hanging leaves, Bushes. Let the Raycast enabled only for the animals.

Let’s run the scene and see what happens. As you can see we are able to select all the animals from the scene. So far, we have animated the animals and added selection function to it for Level 1 GamePlay.

So that's it from this video. Keep practicing and we will meet in the next video!