Drawing Tutorial

This will how to draw lines with your mouse within Unity.

1. **Create a new 2D Project**

   Start with creating a new 2D Project (name can be whatever you like).

2. **Making a Ball**

   Go to the Hierarchy and select the “Create” drop down menu and pick “Create Empty. Or Right-Click and pick “Create Empty”.

   Rename the object to “Ball”. This will be our topmost parent for the ball object.

   Right-Click on the object in the Hierarchy, go to 2D and select sprite. The sprite will become a child object to the parent.

   Move to the “Inspector” and double-click on the small circle icon on the end of the “Sprite” function in the “Sprite Renderer” and Select the “knob” icon. (This can be changed later). Feel free to alter the size of the sprite to fit your game.

3. **Adding a RigidBody2D**

Select the “Ball” Object in the Hierarchy, move back to the “Inspector” and click on the “Add Component” button.

Type in the search bar “Rigidbody2D” and Select said Component.

Find the “Body Type” tab within the “RigidBody2D”, and change it from “Dynamic” to “Static” so it won’t move.

4. **Adding a Collider2D**

   Again, go back to the “Add Component” Button and search for “Circle Collider 2D”.

After adding the Component, look through the “Scene” view to edit the size of the Circle Collider. This can be done by either, changing the “Radius” or Clicking the icon next to “Edit Collider”.

5. **Making a Prefab**

Make a new folder in your Project folder, in “Assets” and name it “Prefabs”.

Drag the “Ball” Object into the “Prefabs” folder.

6. **Creating the Line**

Create a new “Empty Object” and name it “Line”.

Add a Component called “Line Renderer” and change the “width” from anywhere between 0.15 and 0.05. And open the “Positions” tab just above the width tab, and set the “X” value to 5.

Feel free to change the colour through the “materials” options.

Now add another Component called “Edge Collider” and change the “X” value in positions to 0.

And create a prefab of the line, once done you can delete the “Line” from your hierarchy.

7. Scripting

Create a script and call it “DrawLine”. And Before anything we must set out our variables. For this we only need 5. 2 GameObjects, a LineRenderer, a EdgeCollider2D and a List For “Vector2”.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class DrawLine : MonoBehaviour
6   {
7
8       public GameObject linePrefab;
9       public GameObject currentLine;
10
11      public LineRenderer lineRenderer;
12      public EdgeCollider2D edgeCollider;
13      public List<Vector2> fingerPositions;
14
```

We also need to create a new Function in our script, after “void Update”, create one called “void CreateLine.

```
38      void CreateLine()
```

Within this void we are going to write the code that will allow us to draw a line.

```
38  void CreateLine()
39  {
40      currentLine = Instantiate(linePrefab, Vector3.zero, Quaternion.identity);
41      lineRenderer = currentLine.GetComponent<LineRenderer>();
42      edgeCollider = currentLine.GetComponent<EdgeCollider2D>();
43      fingerPositions.Clear();
44      fingerPositions.Add(Camera.main.ScreenToWorldPoint(Input.mousePosition));
45      fingerPositions.Add(Camera.main.ScreenToWorldPoint(Input.mousePosition));
46      lineRenderer.SetPosition(0, fingerPositions[0]);
47      lineRenderer.SetPosition(1, fingerPositions[1]);
48      edgeCollider.points = fingerPositions.ToArray();
49  }
```

MAKE SURE YOU WRITE “fingerPositions.Add(Camera.main.ScreenToWorldPoint(Input.mousePosition));” TWICE OR IT WILL NOT WORK PROPERLY!

Now to make it so we can draw with the “Mouse Pointer”. In the “Update” function you want to write the following.

```
22      void Update()
23      {
24          if(Input.GetMouseButtonDown(0))
25          {
26              CreateLine();
27          }
```

After that, we can move to writing the code for our line to update. We need to create a new function after “CreateLine” and call it “UpdateLine”, within the brackets you want to write “Vector 2 newFingerPos”

```
51      void  UpdateLine(Vector2 newFingerPos)
```

Now we only have a few lines of code to add into this function.

```
51      void  UpdateLine(Vector2 newFingerPos)
52      {
53          fingerPositions.Add(newFingerPos);
54          lineRenderer.positionCount++;
55          lineRenderer.SetPosition(lineRenderer.positionCount - 1, newFingerPos);
56          edgeCollider.points = fingerPositions.ToArray();
57      }
```

Finally for the last bit of code, we need to go back to the “void Update” function and make a new “if” statement.

```
28      if(Input.GetMouseButton(0))
29      {
30          Vector2 tempFingerPos = Camera.main.ScreenToWorldPoint(Input.mousePosition);
31          if(Vector2.Distance(tempFingerPos, fingerPositions[fingerPositions.Count -1]) > .1f)
32          {
33              UpdateLine(tempFingerPos);
34          }
35      }
```

8. **Game Controller**

Go back to Unity and create a new “Empty Object” and rename it to call it “Game Controller”

Apply the “DrawLine” Script onto the Game Controller”. Next drag the “Line” prefab to the “Line Prefab” slot on the “Game Controller.

9. **Run Unity**

Run Unity and try drawing around or draw a slope new to the “Ball” Object and change the “Body Type” from “Static” to “Dynamic”.

Have Fun!