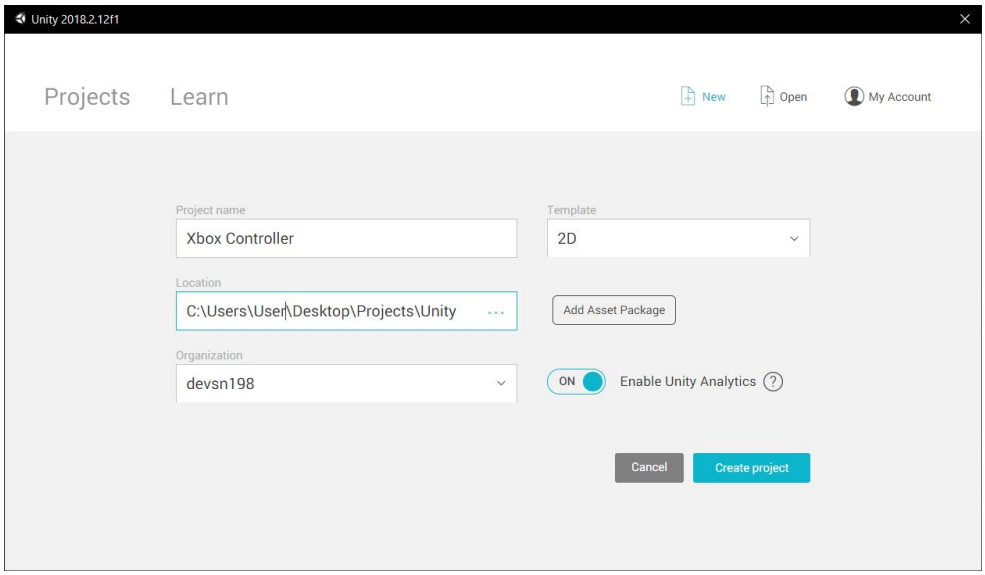
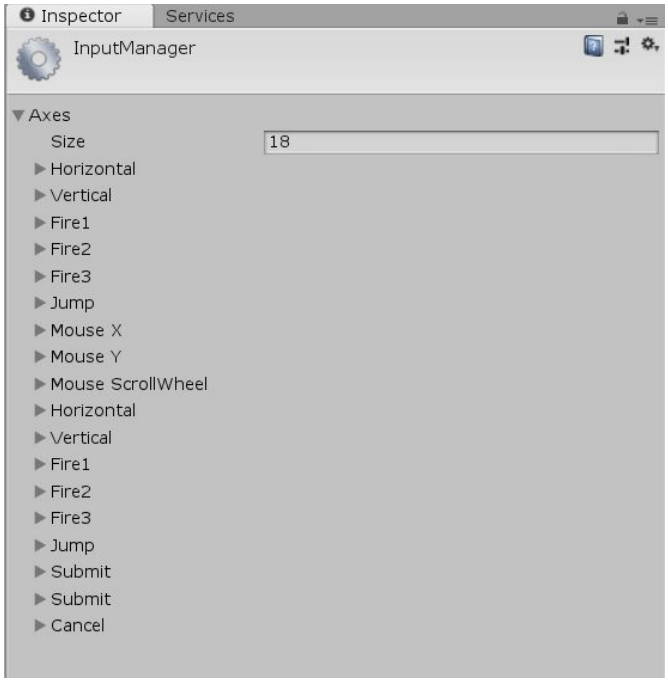
How to Set Up an Xbox 360 Controller and Move a 2D Character in Unity

Tutorial by Dev Soni

1) First let's open up a new 2D Unity File



2) To set up the Xbox controller we must set up its inputs. From the top, click on edit, project settings and then input. In the inspector the input manager will open up. Click on the Axes list.



3) Now based on you are using a Mac or a PC, things will differ slightly. Here is the image from the Unity Xbox360 wikipedia page for reference. More information can be found on the page itself. Here is the link: http://wiki.unity3d.com/index.php/Xbox360Controller

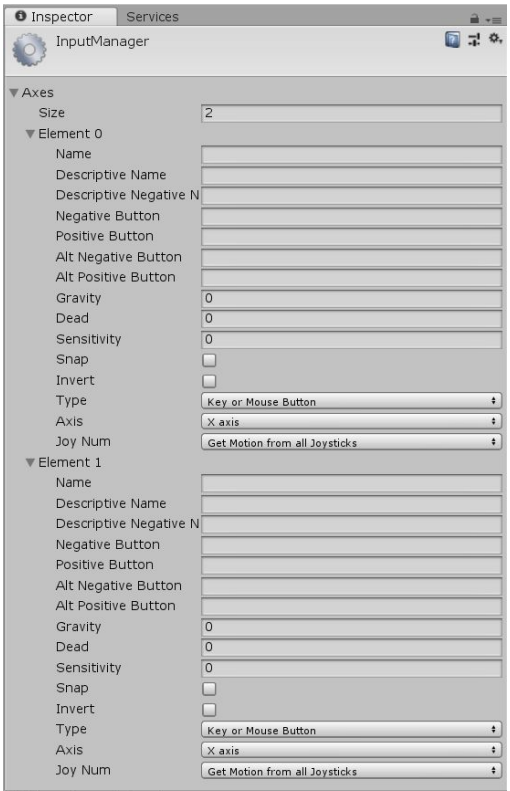
---

Windows

Mac OS X

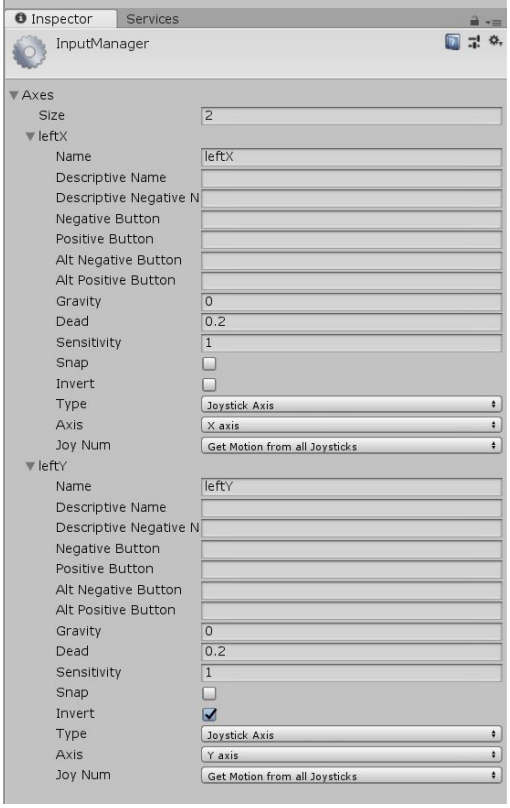
Xbox 360 Controller Layout for Unity
Using Tattie Bogle Mac OSX 360 Controller Driver

4) This tutorial will show just the left joystick being set up but similar steps can be used for any of the inputs on the controller. Change the number 18 in the Axes to 0 and then to 2. This will give us 2 elements to work with. One will be the x input of the left joystick and the other will be the right input of the left joystick. Open them both up. This is what you should see:

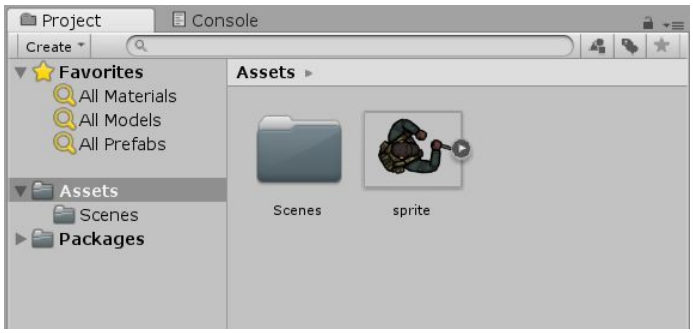


5) Fill out some of the boxes:

- Type a name in the name sections that is appropriate to what the input will be.
- Next set the deadzone to something like 0.2 so that the joystick being slightly off center will not cause movement.
- Set sensitivity to 1.
- Change Type from key or mouse to joystick axis
- Set the axis according to what it is for your OS from the reference image above
- Repeat for the second element but check the invert for the y input.
- The final result should look like the bottom right picture.



6) Now lets drag into Unity a sprite of a game character or anything else that you may want to move around into the assets folder.



3) Drag the sprite into the inspector to create an object and then add a Rigidbody2D component to it by clicking “Add Component” button in the inspector and searching for the component. Set gravity scale to 0 or else your sprite may slowly fall downwards.

4) Next create a new C# script in the inspector and name it something like “MovePlayer” but do not forget to attach it to the sprite object. Here is the script that I have made that will make the player both turn and move.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MovePlayer : MonoBehaviour {

    private Rigidbody2D rb;
    private int speed = 2;
    private float angle;

    void Start() {
        rb = gameObject.GetComponent<Rigidbody2D>(); // Get Rigidbody component
    }
    void Update () {
        Moving(); // Call function in update
    }

    void Moving() {
        rb.velocity = new Vector2(speed * Input.GetAxisRaw("leftX"), speed * Input.GetAxisRaw("leftY")); // Move in direction
        angle =  Mathf.Rad2Deg * Mathf.Atan2(Input.GetAxisRaw("leftX"), -Input.GetAxisRaw("leftY")); // Get look direction
        angle = Mathf.RoundToInt(45 * Mathf.FloorToInt(angle / 45)); // Clamp to 45 degrees intervals
        transform.rotation = Quaternion.Euler(0, 0, angle); // Look in direction
    }
}
```

- By getting the input from the x and y axes, we can tell the sprite’s velocity to move accordingly. If the player’s stick is up then move the player upwards (positive), times the speed for example.
- The angle is found by using a trig function based on what direction the player is moving and then the sprite is rotated accordingly.
- Clamping the angle to 45 degree intervals is optional. If you want the traditional compass 8-direction movement then it is useful, otherwise it is not necessary if you want smooth turning.
- You might have to add + or - 90 or even 180 to the angle if you sprite is not oriented properly. It would look like this:
    - transform.rotation = Quaternion.Euler(0, 0, angle - 90);