CN Controls (v0.5)

Thanks for downloading this CN Controls package. Please read the setup and usage parts carefully. In my experience, most of the problems are caused by improper initial setup.

1. Setup

When you download and import the package, you'll want to add some of the controls on your scene. It uses the standard Unity UGUI system to display the controls, so to add them in your scene, follow these two simple steps:

1. If you don't have a UGUI *Canvas* on your scene yet, add that using **GameObject → UI → Canvas** menu items.

2. Drag any desired control from **CNControls/Prefabs** folder onto your *Canvas*. That's it, you're ready to go.

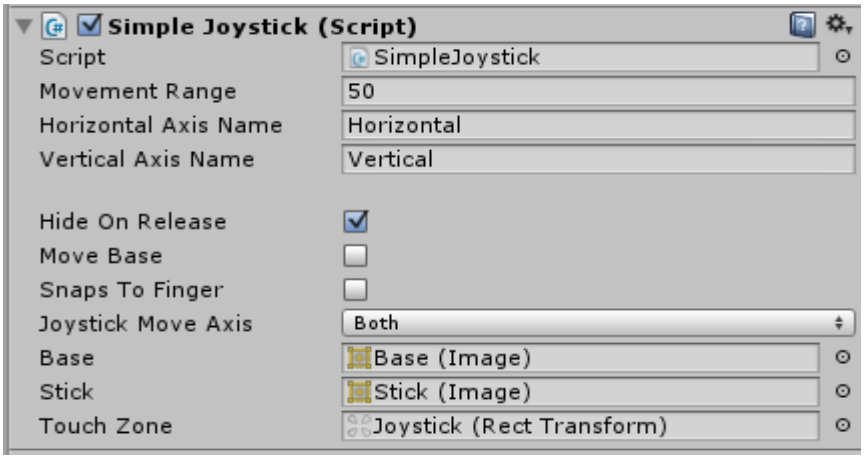
(3). Optional step – tweak the controls position, it's parameters like axis names and such.

The package contains five controls:

1. Joystick
2. Button
3. 4 Way D-Pad
4. 2 Way D-Pad horizontal
5. 2 Way D-Pad vertical

Let's take a look at them one by one.

1.1. Joystick



Properties:

Movement Range – how far can you move the stick of the joystick from its base until it becomes clamped or will make the base move towards it.

Horizontal Axis Name – the name of the horizontal axis of the joystick (which axis should the joystick update with it's X relative position).

Vertical Axis Name – the name of the vertical axis of the joystick (which axis should the joystick update with it's Y relative position).

Hide On Release – whether the joystick should be hidden when it's not tweaked

Move Base – whether the joystick is constrained to its base position (unchecked) or the joystick can move freely across the screen, making its base follow the stick (checked).

Snaps To Finger – whether the joystick should be placed under the finger when it's pressed. *If False, the MoveBase option will be omitted.*

Joystick Move Axis – whether the joystick movement should be constrained by only one axis (vertical or horizontal) or can be moved around freely.

Following three components are not really intended to be changed.

Base – Image component of the Base.

Stick – Image component of the Stick.

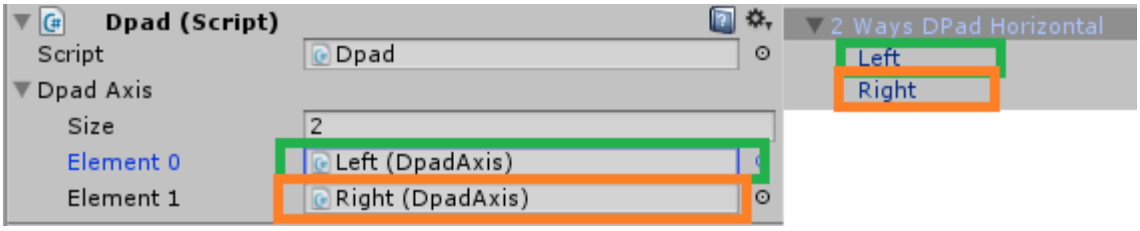
Touch Zone – Rect Transform component of the Active Touch Zone.

1.2. Button

The button has just one property – a button name. Just like in a standard Input system, this is the name of the button you want to query in your game. It supports all the states – *GetButtonDown*, *GetButtonUp* and *GetButton* (whether the button is currently pressed). It's simple as that.

1.3. D-Pad

D-Pad is a bit trickier because almost all of its options are located inside the D-Pad axis and not in the D-Pad itself.



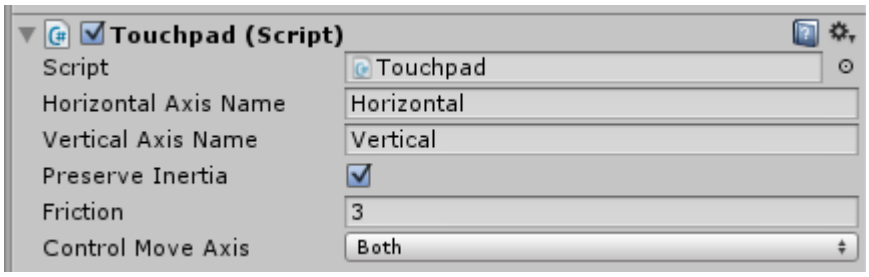
The D-Pad itself only has references to its D-Pad axis which actually provide the Input. They are usually located as children of the D-Pad in the hierarchy (see the pic above). To edit D-Pad axis properties, select the needed one in the hierarchy of the D-Pad.

D-Pad axis properties look like this.

Axis Name – specifies which axis should be updated. D-Pad is much like a joystick, as it updates the Axis values, not the Button states.
Axis Multiplier – the value on which the axis will be updated when the D-Pad axis will be pressed. For example, -1 multiplier for the Left D-Pad axis will update the *Horizontal* Axis value to -1. Take a note thought that if this value is greater than 1 or less than -1, it will be clamped to [-1, 1] (matter of opinion here, feel free to write your preferred usage, it can be tweaked in the future).

1.4. Touchpad

Touchpads are usually used with camera control or map scrolling.



Horizontal Axis Name – the name of the horizontal axis of the touchpad (which axis should the touchpad update with it's X relative position).

Vertical Axis Name – the name of the vertical axis of the touchpad (which axis should the touchpad update with it's Y relative position).

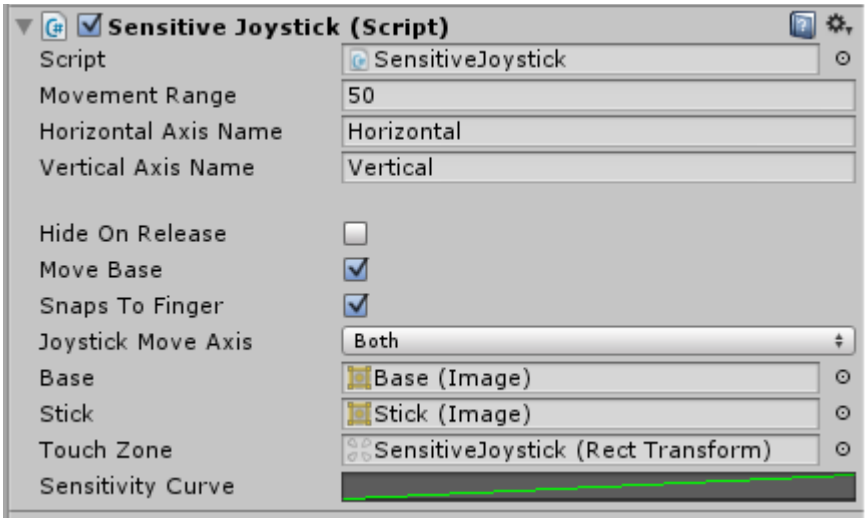
Preserve Inertia – whether the touchpad should continue to submit axis values to the input system even when it's not being tweaked (when the user lifts his/her finger from the screen). In the example scene with the camera, it looks like the camera movement slowly fades and finally stops.

Friction – how fast should the touchpad fade it's values once the user stopped tweaking the touchpad or stopped scrolling. The bigger the value, the faster the touchpad values will fade.

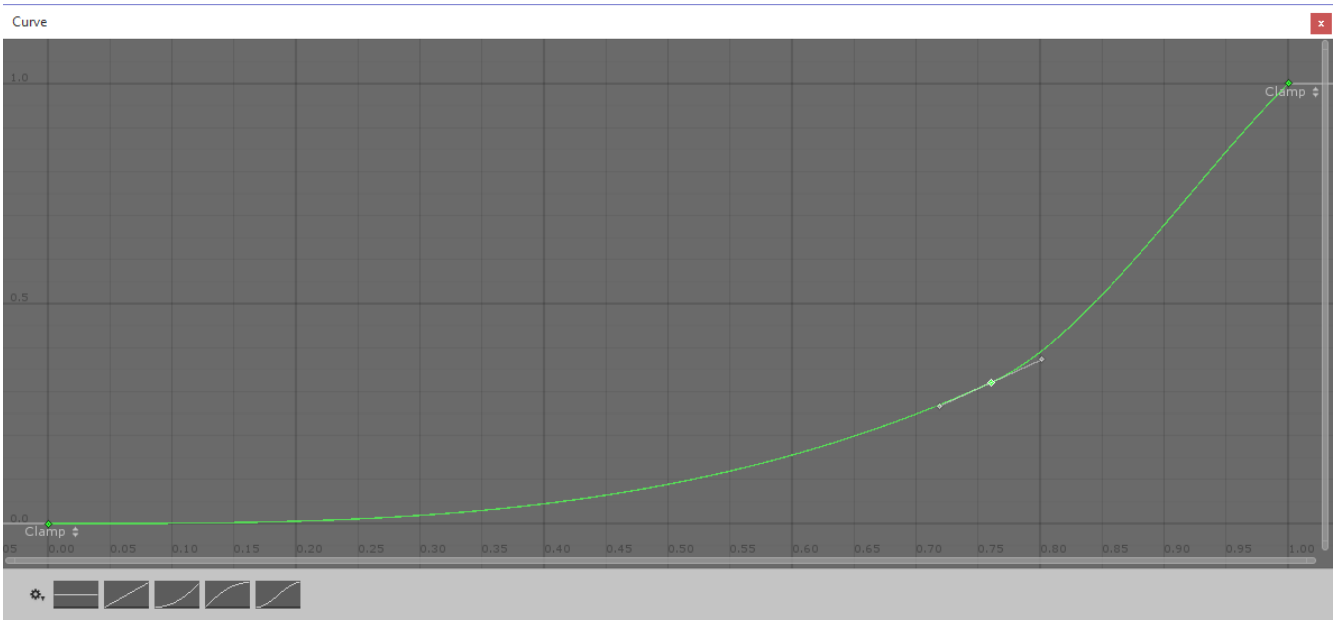
Control Move Axis – should the touchpad change the values of X, Y or Both axis. Similar to Joystick Move Axis, but only affects the values of the axis as there are no moving controls.

1.5. Sensitive Joystick

Sensitive Joystick sets up and acts like a normal joystick, but in addition it has a sensitivity curve which allows you to precisely tweak the input curve of the joystick.



Most of the properties are identical to the Joystick (See 1.1. Joystick), but there's also an additional **Sensitivity Curve** property, which is linear by default. Double clicking on it opens the Unity Curve Editor in which you can tweak the “keyframes”, making your ultimate precise joystick.



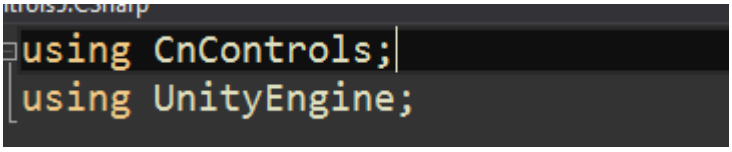
The joystick input values will be fed into this curve, sampling the actual value. Note that joystick only feeds the 0-1 values, clamping all the values that are not in this range – its default keyframes are at (0,0) and (1,1) values.

2. Custom images

To change the look of your controls, replace the graphics with yours and optionally tweak the size. The graphics themselves are simple UGUI Image components, so you can change them to any Sprite texture.

3. Usage

The usage is pretty simple. To use *CnControls*, you'll need to add a "*using CnControls*" line at the beginning of a script file:



And then, in any part of the code, query the *CnInputManager* like you would normally do with standard *Input*. Like *CnInputManager.GetAxis("Horizontal");*

```
Vector3 movement = new Vector3(CnInputManager.GetAxis("Horizontal"),
    CnInputManager.GetAxis("Vertical"), 0f);
```

And that's it, you're ready to go.

You can also check the examples included in the package. It has a 2D platformer example and a simple third person controller.

4. Playmaker support

There's an optional "Playmaker Actions" package that contains all the needed Actions to use this system with Playmaker. It's not free, but it's as cheap as it can be. You can find it here http://u3d.as/iqc. Purchasing this package is a great way of supporting this CN Controls project.

5. Useful links

- If you have any questions, feel free to drop me a line at cyrill@nadezhdin.org
- Check out my blog at http://blog.nadezhdin.org/
- The best way to support this project is to check out my other packs at https://www.assetstore.unity3d.com/en/#!/publisher/6074/