Hex Gamepad Controller

Documentation

Hex Gamepad Controller Version:1.0
Documentation version: 1.0

TABLE OF CONTENTS

Overview: p.1
Setup (Universal) p.2
Setup: (Turn Based Strategy Framework) p.4
Using the Mouse: p.9
Using the Game Controller: p.10
Using the Keyboard: p. 11
Working with the Scripts: p. 12
Additional Information: p.13

OVERVIEW

Hex Gamepad Controller (HGC) lets you use a joystick or game pad (such as an Xbox 360 controller) or the keyboard with any 3D hex-based map. It's perfect for turn-based hex games at a fraction of the price of other controller assets.

In addition to its universal functionality, it has a custom-designed option to work with the excellent "Turn Based Strategy Framework" (TBSF) by Michał Żętkowski (see https://assetstore.unity.com/packages/templates/systems/turn-based-strategy-framework-50282).

This package contains two sets of scripts, one for universal use and the other for use with TBSF. The universal version is standalone; the TBSF version requires you to make included additions to TBSF scripts (and of course you'll need to buy TBSF).

Features:

- Works with any 3D hex-based map
- TBSF-specific code works out of the box with minimal setup
- Includes free cursor and hex prefabs
- Automatic edge detection to keep the gamepad cursor on any map automatically
- "Chunky" movement so the gamepad cursor snaps from one hex to the next while moving
- Support for selection of units on the map, including unit cycling
- Full camera support: follow the gamepad cursor, zoom in/out, rotate, vertical pan.
- Full support for gamepad, keyboard and mouse
- Fully commented and easy to connect to your existing game
- Works with Unity 2017

Please note that HGC currently only works with hex maps in flat top orientation, though it can be easily modified to work with pointy top maps.

SETUP

Universal HGC

This will show you how to set up HGC for any 3D hex-based map.

1. Delete the folder "ForTBSF". Keep the "Common" and "Universal" folders.

*(**Note**: if you want to just skip steps 1-6, feel free to just use the included demo scene as your starting point.)*

2. In the Hierarchy, create a parent game object that contains the hexes in your map. For the purposes of these instructions, we'll call it "Grid" but any name will do.

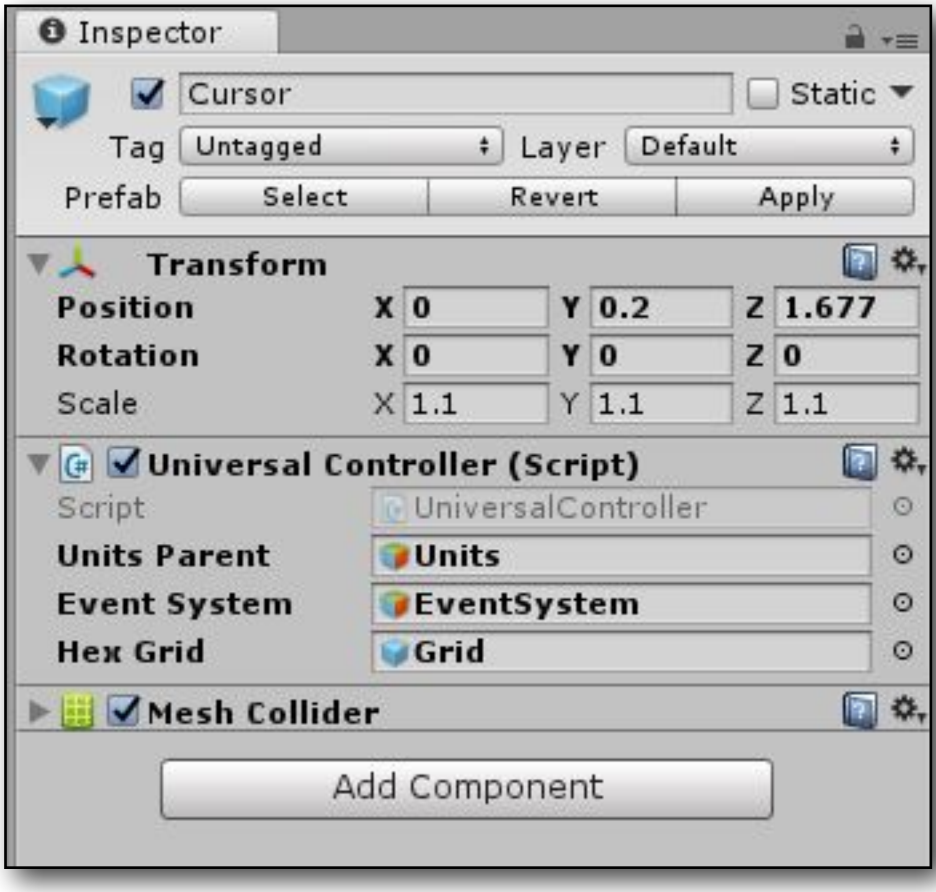
3. In the Hierarchy, create a parent game object that contains the units in your game. For the purposes of these instructions, we'll call it "Units" but any name will do.

4. In the Hierarchy, if you don't have an EventSystem already, create one.

5. In the Hierarchy, add the "Cursor" by dragging it to the Hierarchy from the Universal/Prefabs folder. In the attached Universal Controller script, there are three variables that need to be populated:

For "Units Parent", drag the "Units" game object (in the Hierarchy) over to it.
For "Event System", drag the "EventSystem" object over to it.
For "Hex Grid" drag "Grid" over.

When you're done, the Cursor in the Inspector should look like this:



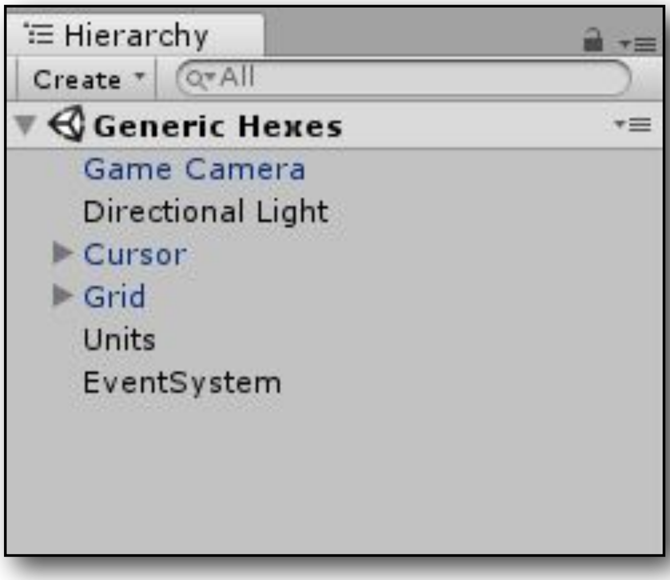
Cursor
Tag: Untagged Layer: Default

	X	Y	Z
Position	0	0.2	1.677
Rotation	0	0	0
Scale	1.1	1.1	1.1

Script: UniversalController
Units Parent: Units
Event System: EventSystem
Hex Grid: Grid

6. In the Hierarchy, delete the "Main Camera" object and replace it with the "Game Camera" (in Universal/Prefabs). In the "Camera Control" script, drag the Cursor over to populate the Target variable. This will let the camera follow the cursor.

Your Hierarchy should now look like this:



7. Create a new layer, Layer 8, and call it "Hex". Then create Layer 9, call it "Friendly" and Layer 10, "Enemy".

(Note: if you want to use different layer numbers, you can do so but also change them within the "Layer" script (found in Common / Scripts. This script also contains layers for friendly and enemy units if you wish to use those).

8. In the Hierarchy, open the "Grid" game object. Select all the hex objects within it. First, assign the "UniversalHex" script to all of them. Second, using the Layer dropdown menu, assign the "Hex" layer to all of them. When asked if you want to assign it to their children, say yes.

9. In the Hierarchy, open the "Units" game object. Select all of the units within it. First, assign the "UniversalUnit" script to all of them. Second, using the Layer dropdown menu, assign the "Friendly" layer to all of them (feel free to use the "Enemy" layer as appropriate). When asked if you want to assign it to their children, say no.

10. Installation is now complete. You might also want to refer to the included demo scene for comparison. Also, see below for details on how to work with the scripts.

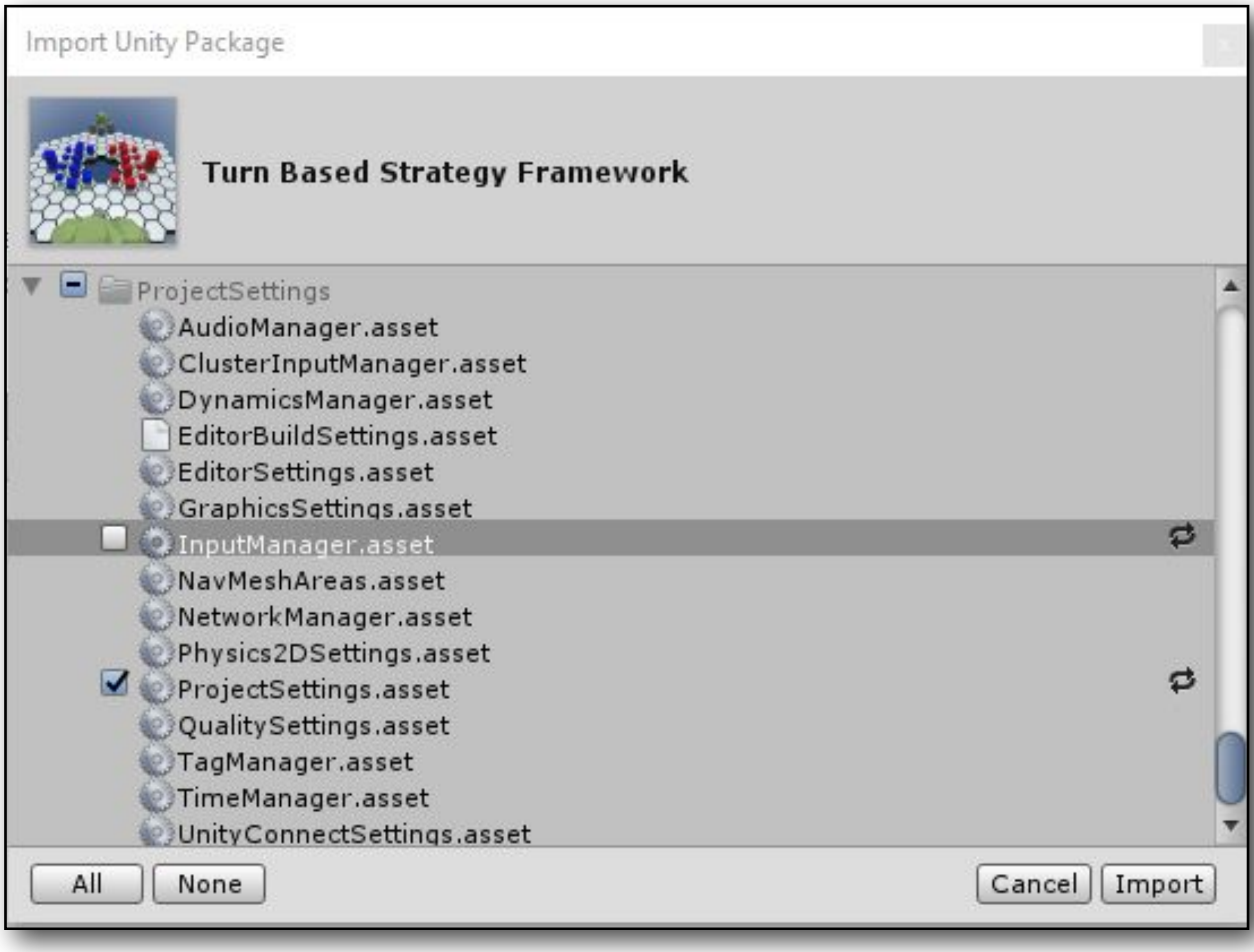
Turn Based Strategy Framework HGC

This will show you how to set up HGC for the "Example Scene 1" scene in the Turn Based Strategy Framework (TBSF). You can do the same thing for other scenes, as long as they are 3D. So, before you start, make sure you are in "Example Scene 1".

1. Delete the folder "Universal". Keep the "Common" and "ForTBSF" folders.

2. Download and import the Turn Based Strategy Framework (TBSF) asset from the Unity Asset Store.

Important: when importing, deselect the "InputManager.asset" (in the Project Settings folder in the import list (see illustration below). This will avoid it overwriting the HGC custom input manager.


Import Unity Package

Turn Based Strategy Framework

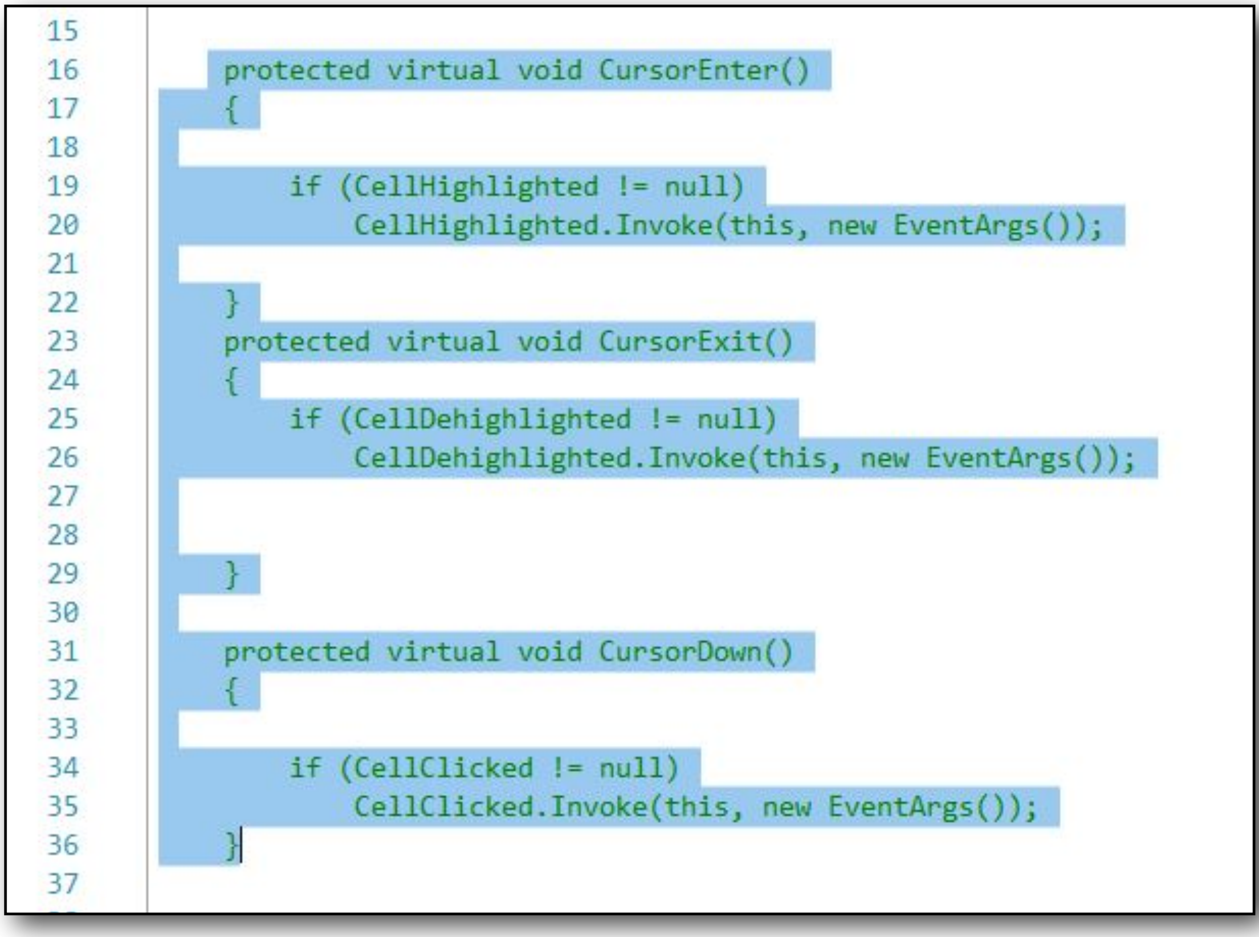
ProjectSettings
AudioManager.asset
ClusterInputManager.asset
DynamicsManager.asset
EditorBuildSettings.asset
EditorSettings.asset
GraphicsSettings.asset
☐ InputManager.asset
NavMeshAreas.asset
NetworkManager.asset
Physics2DSettings.asset
☑ ProjectSettings.asset
QualitySettings.asset
TagManager.asset
TimeManager.asset
UnityConnectSettings.asset

3. In the imported TBSF asset, open the following scripts:

> Cell
> MyHexagon
> Unit
> MyUnit

4. In the "HGC / ForTBSF" folder, open the "AddOns" script. You will see commented code for each of the four above TBSF scripts. Within "AddOns", copy and paste the code for the relevant TBSF script into the class script, then de-comment it.

As an example of how to do this, in the "Cell" section of "AddOns", copy the code there, paste it into the "Cell" script, and uncomment it:



```

    protected virtual void CursorEnter()
    {

        if (CellHighlighted != null)
            CellHighlighted.Invoke(this, new EventArgs());

    }
    protected virtual void CursorExit()
    {
        if (CellDehighlighted != null)
            CellDehighlighted.Invoke(this, new EventArgs());


    }

    protected virtual void CursorDown()
    {

        if (CellClicked != null)
            CellClicked.Invoke(this, new EventArgs());
    }

```

Note: The "Unit" script has several classes in it. To be certain to paste the code into the right class, paste it into line 131 just below the OnMouseExit function.

Important: You will also need to change the access level of the OnMouseDown method in "Cell" to "protected virtual void" so that mouseDown events will work on hexes.

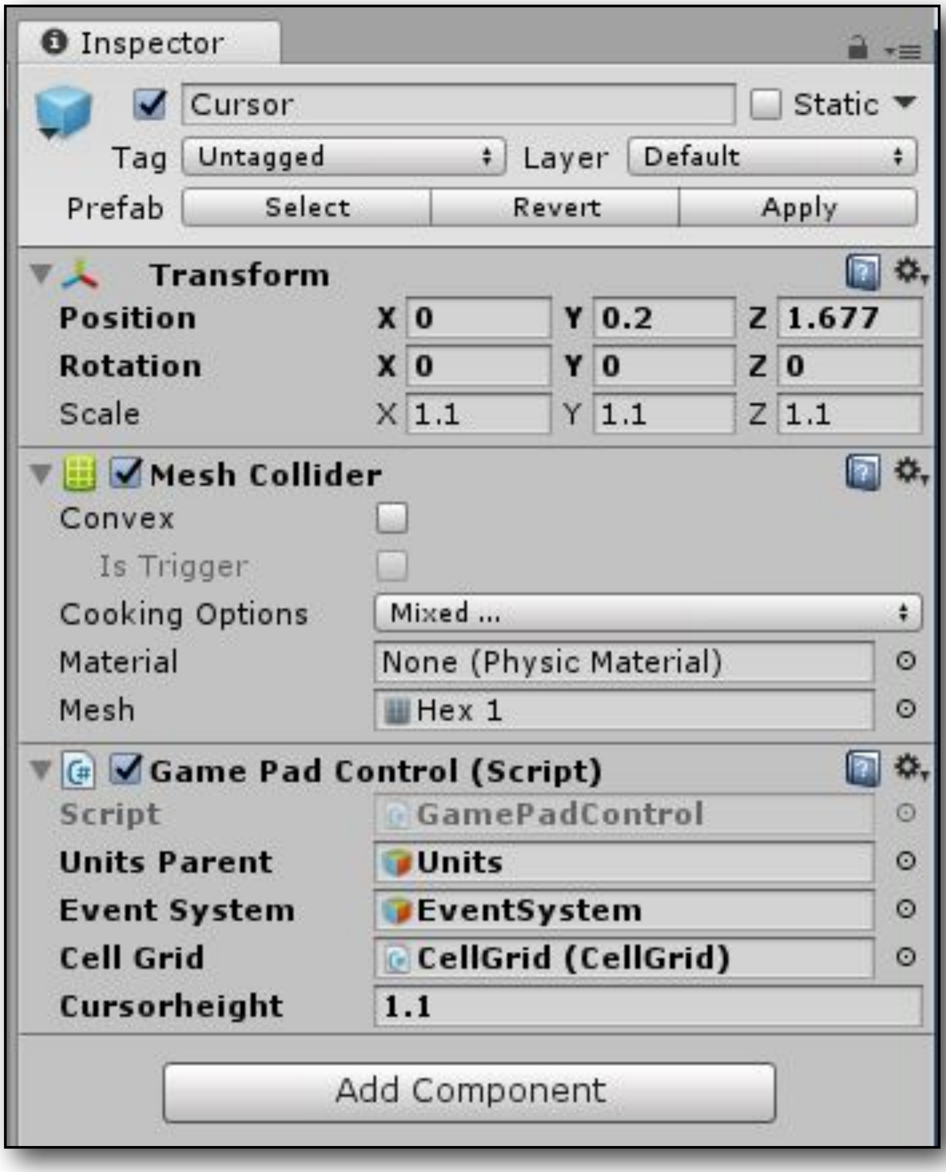
5. Open the GamePadControl and CameraControl scripts, and uncomment them. In GamePadControl, also uncomment the two "using TBSFramework...." lines.

6. Open the "Example Scene 1" scene in TBSF (in TBS Framework / Scenes).

7. In the Hierarchy, add the "Cursor" by dragging it to the Hierarchy from the ForTBSF/Prefabs folder. In the attached Universal Controller script, there are three variables that need to be populated:

> For "Units Parent", drag the "Units" game object (in the Hierarchy) over to it.
> For "Event System", drag the "EventSystem" object
> For "CellGrid" drag "CellGrid" over.

When you're done, the Cursor in the Inspector should look like this:



Cursor ✓
Tag: Untagged
Layer: Default

Transform
Position X 0 Y 0.2 Z 1.677
Rotation X 0 Y 0 Z 0
Scale X 1.1 Y 1.1 Z 1.1

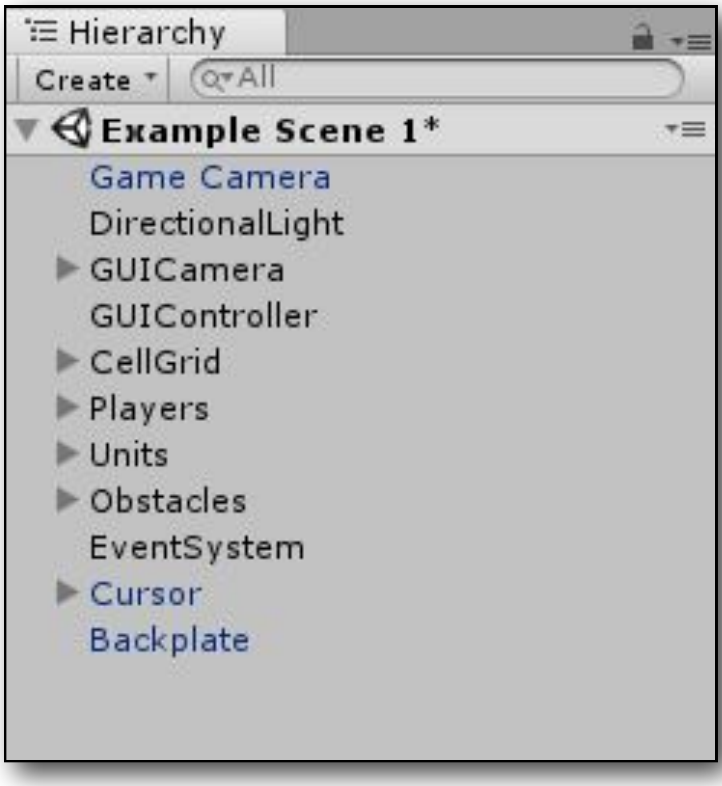
✓ Mesh Collider
Convex ☐
Is Trigger ☐
Cooking Options: Mixed ...
Material: None (Physic Material)
Mesh: Hex 1

✓ Game Pad Control (Script)
Script: GamePadControl
Units Parent: Units
Event System: EventSystem
Cell Grid: CellGrid (CellGrid)
Cursorheight: 1.1

8. In the Hierarchy, delete the "Main Camera" object and replace it with the "Game Camera" (in ForTBSF/Prefabs). In the "Camera Control" script, drag the Cursor over to populate the Target variable. This will let the camera follow the cursor.

9. Drag the "BackPlate"object (in ForTBSF / Prefabs) to the Hierarchy. This object provides a ray casting target for any missing hexes in the interior of the map (as in Example 1) so it doesn't interfere with zooming and mouse edge scrolling.

Your Hierarchy should now look like this:



10. Create a new layer, Layer 8, and call it "Hex". Then create Layer 9, call it "Friendly" and Layer 10, "Enemy".

(Note: if you want to use different layer numbers, you can do so but also change them within the "Layer" script (found in Common / Scripts. This script also contains layers for friendly and enemy units if you wish to use those).

11. In the Hierarchy, open the "CellGrid" game object. Select all the objects within it. These are labelled "Hexagon(Clone)". Using the Layer dropdown menu, assign the "Hex" layer to all of them. When asked if you want to assign it to their children, say "yes".

12. In the Hierarchy, open the "Units" game object. Select all of the units within it. Using the Layer dropdown menu, assign the "Friendly" layer to all of them; the Enemy layer is for your use later at your discretion). When asked if you want to assign it to their children, say "no".

13. Installation is now complete. Also, see below for details on how to work with the scripts.

USING THE MOUSE

Use of the mouse is the same as in regular TBSF in terms of selecting units and hexes and clicking on UI elements (i.e. using the left mouse button).

The mouse controls the camera in the following ways:

Mouse pointer: When it hits the edge of the screen, it scrolls the screen in that direction until it hits the edge of the map. Note that the map has to occupy at least half of the screen for this to work (or the mouse will think it's gone off the edge and stop scrolling).

Right mouse button: Hold to rotate the view

Middle mouse button: Hold to pan around the map

Mouse scrollwheel: Zoom In / Out

If you want to switch to the controller, press the "M" key.

USING THE GAME CONTROLLER

HGC is set up for an Xbox controller. If you want to use a different controller, you'll have to make the appropriate settings in InputManager for the specific controller you want to support (the settings for different controllers can be found on the web). Use the "M" key to switch to the mouse. The Xbox settings are here:

Submit	A
Cancel	B
End Turn	Y
Cycle Unit Up	RB
Cycle Unit Down	LB
Move Cursor	L
Move Camera	R
Zoom In	RT
Zoom Out	LT

USING THE KEYBOARD

Submit	Enter
Cancel	ESC
End Turn	Space
Cycle Unit Up	E
Cycle Unit Down	Q
Move Cursor	WASD
Rotate Camera	Number Pad: 4 and 6
Nudge Camera up and down	Number Pad: 8 and 2
Zoom In	Number Pad: + (plus sign)
Zoom Out	Number Pad: - (minus sign)

WORKING WITH THE SCRIPTS

Both the Universal and TBSF versions of HGC have a similar script structure:

- Cursor controller (either "UniversalController" or "GamePadController", depending): This controls the cursor, edge detection, unit selection and so on.
- Camera (either "UniversalCamera" or "CameraController") which controls the game camera.
- Parent class script for hexes (either the Universal mode's "HexActions" or the TBSF script "Cell"): These are the scripts that do things when a mouse or cursor event happens on the hex (i.e. it is selected, the pointer or cursor moves over it, etc).
- Parent class script for units (either the Universal mode's "UnitActions" or the TBSF script "Unit"): These are the scripts that do things when a mouse or cursor event happens on the unit (i.e. it is selected, the pointer or cursor moves over it, etc).
- Child class script for hexes (either the Universal mode's "UniversalHex" or the TBSF script "MyHexagon"): For the purposes of HGC, these scripts interpret mouse or controller/ keyboard inputs into events for their parent class (though of course TBSF's "MyHexagon" does a lot more).
- Child class script for units (either the Universal mode's "UniversalUnit" or the TBSF script "MyUnit"): For the purposes of HGC, these scripts interpret mouse or controller/keyboard inputs into events for their parent class (though of course TBSF's "MyUnit" does a lot more). Also, the units in the TBSF scene "Example 1" have further type-specific child scripts that call up to "MyUnit".

(The reason I have kept mouse and controller/keyboard events separate is so that some things only happen when the mouse is active, and others when the controller/keyboard is. I found that having them both active at the same time ran into too many problems.)

The TBSF version also has an "AddOns" script, which as described above is really just a text file used for copying and pasting code as part of installation.

For both the Universal and TBSF scripts, put code that affects the hex or unit being selected (e.g. highlighting it) into the child class script. Code that interacts with the rest of your game engine (when a hex / unit is interacted with) should go into the parent class scripts. I've put some simple example code in the Universal scripts.

For the Camera and Cursor settings, please refer to the comments in their respective scripts for information on what the various settings do.

Finally, all the HGC scripts are well-commented, but should you have any questions that appear to be immune to Googling, please feel free to get in touch at info@foiblegames.com . Please note that I can only help you with the HGC scripts and their interaction with TBSF. For help with TBSF itself, please contact the developer.

ADDITIONAL INFORMATION

Developer website: http://www.foiblegames.com/

Support email: info@foiblegames.com

Credits:

Thanks to Kira for the Blender magic on the hexes and cursor.

Thanks to Michał Żętkowski, creator of TBSF, for his support. Check out TBSF at https://assetstore.unity.com/packages/templates/systems/turn-based-strategy-framework-50282).

Xbox controller button images courtesy of lukezammit at https://www.titanui.com/49580-xbox-one-controller-buttons-psd/