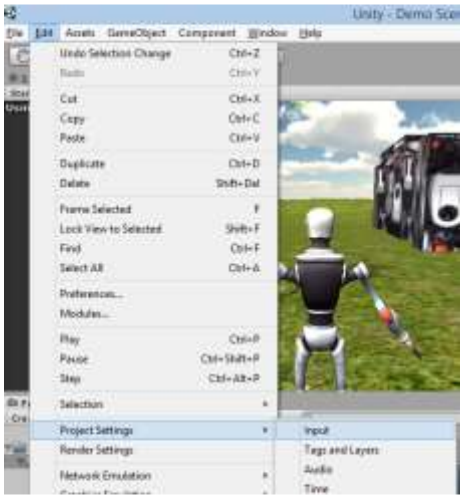
Essential RPG Controller

By: Stand Off Software, a Brindle Waye company

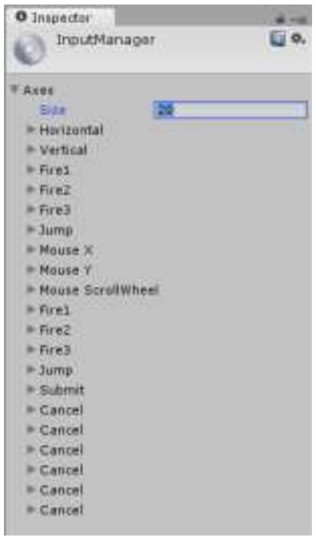
Contact: game.support@brindlewaye.com

Controller Setup:

Five new input axes must be added to the project settings input manager. To open the input manager, select “Edit” from the Unity main menu, then select “Project Settings” and “Input” from the popup menus that are presented.



This will bring up the InputManager inspector, as shown here:



Since we are adding five new items into this list of input axes, first add 5 to the number shown in the “Size” field. Press enter and see five new entries at the bottom of the list. (This step has already been done in the screen shot shown here.)

Initially these new list entries will be a copy of the previous last item in the list. Therefore, my list now shows six input axes named “Cancel.” Starting with the first new Cancel input, click on each one in order and modify to the new values shown here.

1. Mouse 0 – Set the Name to “Mouse 0” and the Positive Button value to "mouse 0"
2. Mouse 1 – Set the Name to “Mouse 1” and the Positive Button value to "mouse 1"
3. Run – Set the Name to “Run” and the Positive Button value to "left shift" (You may use another value of your choosing)
4. Strafe – Set the name to “Strafe” : Set the Positive Button and Negative Button values to your liking; the standard is positive "e" and negative "q"
5. ModeChange – Set the name to “ModeChange” : Set the Positive Button value to whatever you want. We suggest "b"

To test the system in your own game:

Set up the inputs as described above, and add the "Everything" prefab (Prefabs folder) to your scene. This will give you the same functionality as the demo with the robot avatar included and some sample animations. Note that in the "Everything" prefab, the canvas and the avatar object are parented to a game object called "Everything." This is **not** the recommended structure. It was just done that way so that everything could be added at once with one prefab to test it out.

To apply the system manually to your own game:

1. Add the inputs as described above. You may leave out the "ModeChange" input if you don't intend to have this functionality.
2. Add the Essential Control Script to your player avatar. (from the "Scripts" folder)
3. Set one of the animation controllers from the "Animation Controllers" folder as the controller for your avatar's Animator component.
4. Add your animations to the states of the animation controller you have chosen (The Simple version has some animations included, but they are only there as an example and are not production quality)
5. Parent the Camera Rig prefab ("Prefabs" folder) to your player avatar.
6. In the Camera Manager script on the Camera Rig, set the Character Object to your player avatar game object.
7. Optionally set a crosshair image on the Camera Manager script.
8. Add the Controller Canvas prefab ("Prefabs" folder) to your scene. If you already have a canvas, move the Screen object from that canvas to your own canvas and remove the Controller Canvas
9. Make sure the "Screen" panel covers the entire screen. Also make sure it is drawn beneath any UI elements that need to receive mouse input.
10. On the event trigger component on the Screen object, set the game object to which you added the Essential Control Script and the Camera Rig. Set the event handler for each to EssentialControlScript.OnPointerDown and CameraManager.OnPointerDown
11. At this point, you may want to remove any other Main Camera you had in your scene.

The Camera Rig consists of a few different parts:

1. Camera Rig object – This object is the parent of all of the other objects and your player avatar should be its parent. It has the Camera Manager on it. It's best if its position stays at {0, 0, 0}. Its rotation **must** stay at {0, 0, 0}.
2. Horizontal Joint – This is the point around which the camera rotates horizontally. You may adjust its position as you like, but its rotation **must** stay at {0, 0, 0}.
3. Vertical Joint – This is the point around which the camera rotates vertically. You may adjust its position as you like, but its rotation **must** stay at {0, 0, 0}.
4. No Clip Target – The camera will move such that there are no colliders in between it and this object. You may adjust its position as you like. Its rotation doesn't really affect anything.
5. Main Camera – This is the main camera. You may adjust its position and rotation as you like.

Camera Manager Script options:

- Character Object – This is the player avatar. The Camera Rig should be parented to this object.
- No Clip Target – This should be set to the No Clip Target object.
- Horizontal Joint – This should be set to the Horizontal Joint object.
- Vertical Joint – This should be set to the Vertical Joint object.
- Main Camera – This should be set to the Main Camera object.
- Find Renderers – If true, the script will find all renderers parented to the Character Object to use for the fader. In order for the fader to work, the material **must** use a transparent shader.
- Renderers – If Find Renderers is not set to true, the fader will only use renderers in this list. In order for the fader to work, the material **must** use a transparent shader.
- Fade Distance – How close to the No Clip Target the camera has to be before starting to fade out the renderers in the Renderers list (which will be filled automatically if Find Renderers is checked). Once again, in order for the fader to work, the material **must** use a transparent shader.
- Hide Distance – How close to the No Clip Target the camera should be before it disables the renderers in the Renderers list (which will be filled automatically if Find Renderers is checked).
- Mode – RPG mode works like a standard RPG camera and FPS mode works like a standard FPS camera.
- Crosshair – If set, this image will be enabled in FPS mode and disabled in RPG mode.
- Mouse Sensitivity – How responsive the camera rotation is to mouse movement.
- Bottom Rotation Extent – How far down the camera is allowed to rotate vertically.
- Top rotation Extent – How far up the camera is allowed to rotate vertically.
- Snap Speed – How fast the camera returns to its original horizontal rotation when the Character Object starts moving.
- No Clip Speed – How fast the camera moves to avoid clipping.
- Zoom Speed – How fast the camera moves when using the scroll wheel to zoom.
- Zoom Amount – How far one notch of the scroll wheel moves the camera in or out.

- Forward Zoom Extent - How far forward the camera is allowed to move.
- Backward Zoom Extent - How far backward the camera is allowed to move.

Essential Control Script options:

- Animation Speed – Set the speed of the animator.
- Turn Speed – How fast the character turns.
- Fall Height – How high off the ground the character must be before playing the falling animation.
- Run Is Toggle – This specifies whether the run button should be treated as a toggle. If false, the button must be continuously held down to run.
- Run After Fall – This specifies whether the run toggle should continue to be on after a fall as long as there is input from the vertical axis. If Run is not a toggle, this option does nothing.
- Mode – RPG mode works like a standard RPG controller and FPS mode works like a standard FPS controller.

Screen Panel:

The purpose of this panel is so that any UI objects in front of it will block mouse input from getting to the camera and control scripts which allows for UI interaction without inadvertently moving the character or camera. It should cover the entire screen and have an opacity of 0.

The following folders are included in this package:

- Animation Controllers – This folder contains two animation controllers. They both work with the character controller, one just has more states for things like starting and stopping and so forth. Choose whichever one fits your animations and project.
- Images – This folder contains the sample crosshair sprite.
- Prefabs – This folder contains the Prefabs: "Controller Canvas," "Camera Rig," and "Everything."
- Scripts – This folder contains the scripts "Camera Manager" and "Essential Control Script." It also contains a test script for switching modes at runtime, but you will probably want to put this code somewhere else rather than use this script.
- Shaders – This folder contains a shader called "VertexLit with Z." It solves the Z issues with transparent shaders, but is a VertexLit shader, so you may want to find a different transparent shader with Z for production depending on your game. This shader came from a free package from Unity. We did not write it, and do not support it.
- Demo Scene – a folder containing the demo scene and its assets.

Support:

If you have any questions, do not hesitate to contact us at game.support@brindlewaye.com.