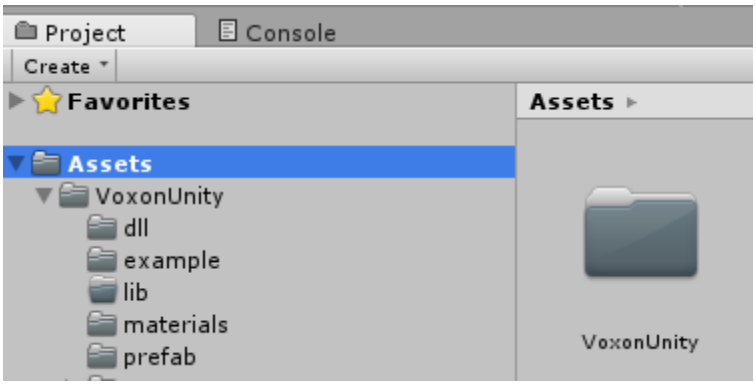
Utilizing Voxon Unity Plugin 0.1

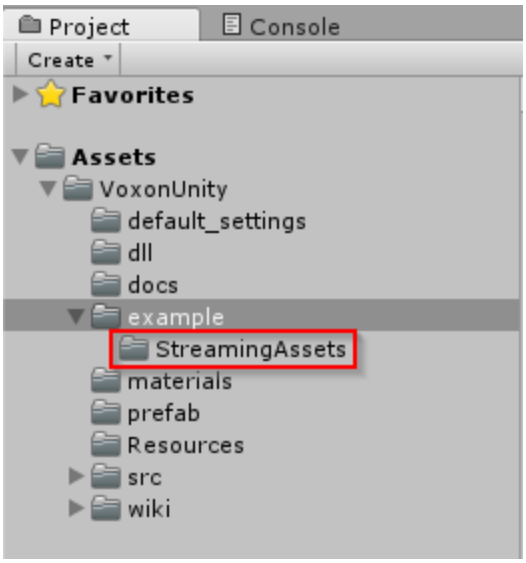
UNITY EDITOR

Create a new project.

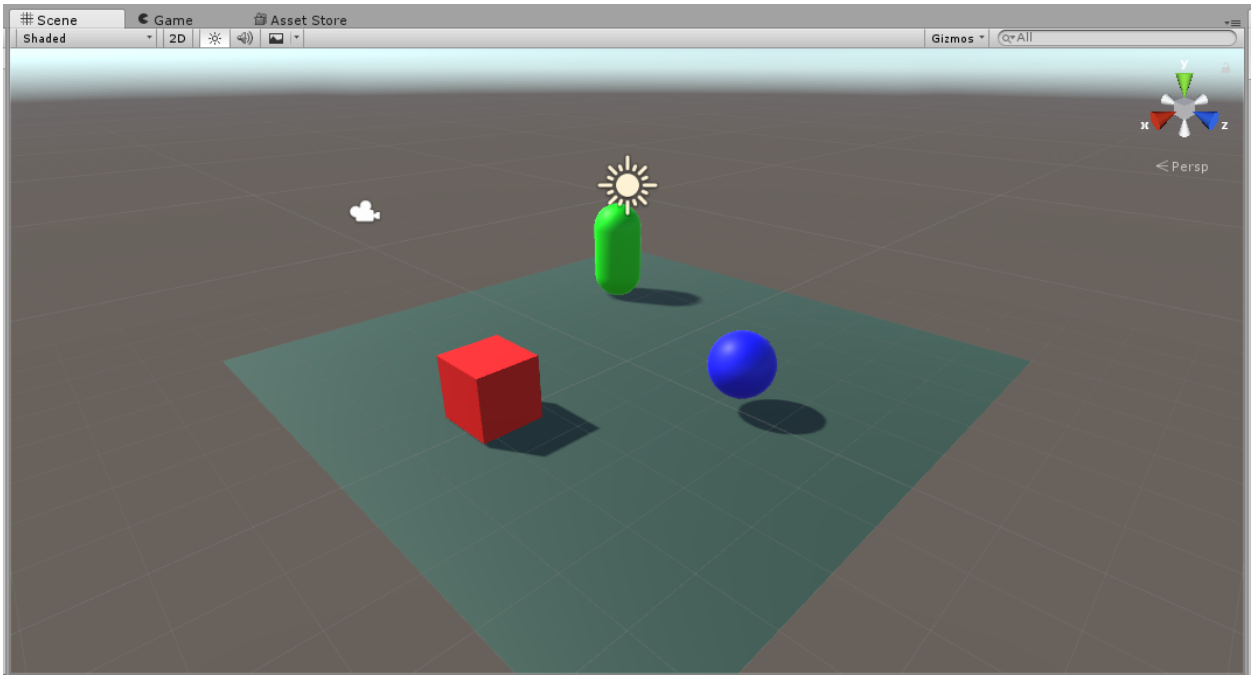
Import the Voxon Unity plugin into your scene by dropping the supplied folder onto the project's Assets folder.



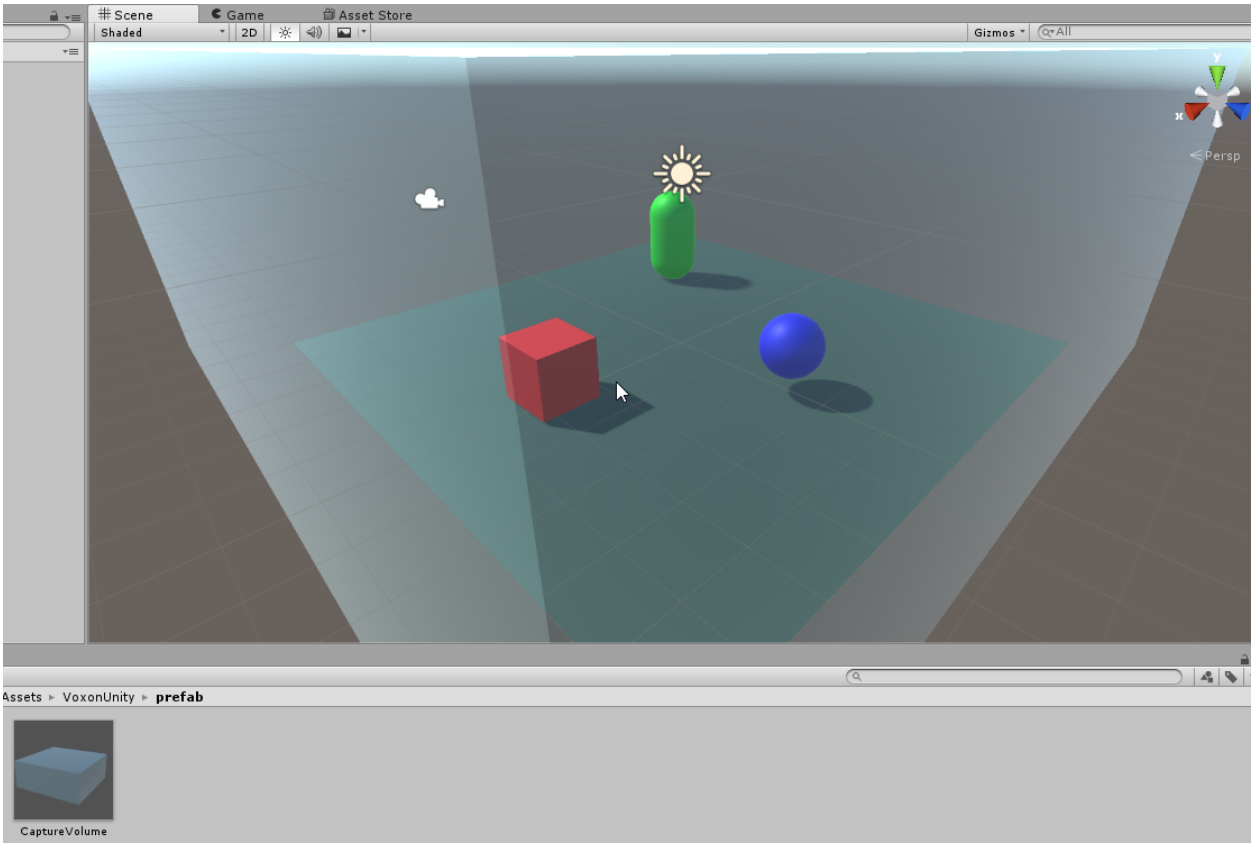
The Voxon Unity plugin requires a StreamingAssets folder in Assets to store and load input configuration. Either create one or move the folder provided in the Assets/VoxonUnity/example folder.



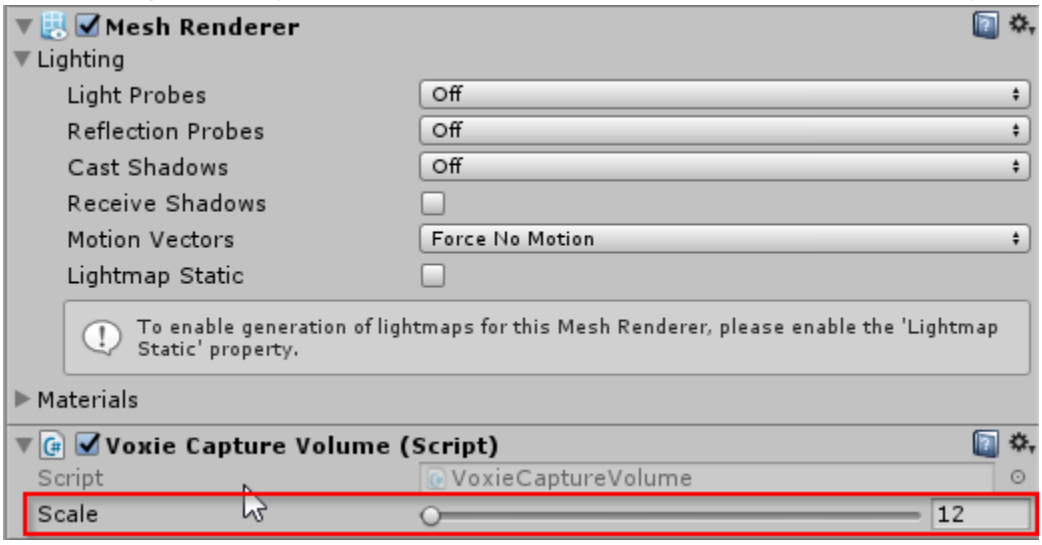
Create a simple scene (suggestion, a plane and a few objects with different colour materials to fill the scene).



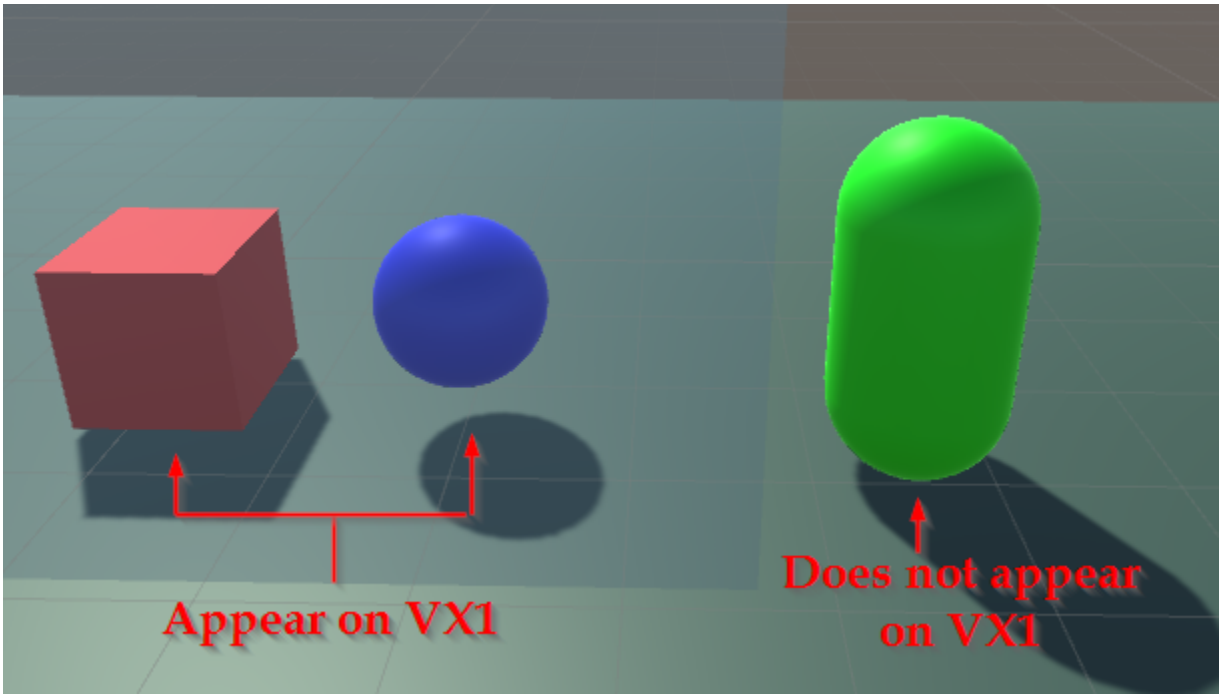
Drag the *CaptureVolume* prefab from *Assets/VoxonUnity/prefab* into your scene.



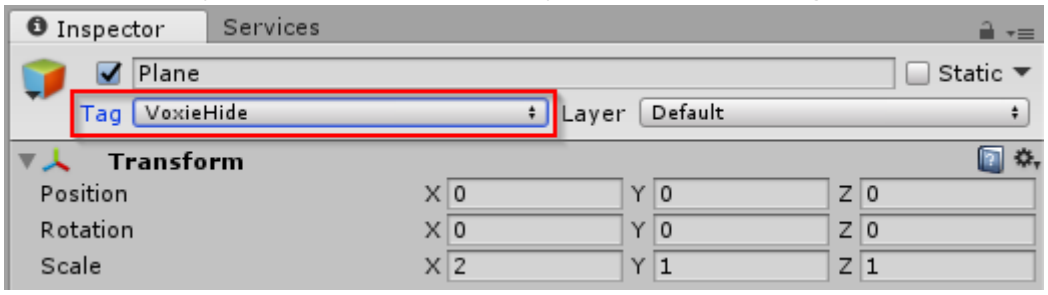
If it is too large / small, you can rescale the volume with the scale slider on the Inspector



Ensure anything you wish the be draw falls within the CaptureVolume.

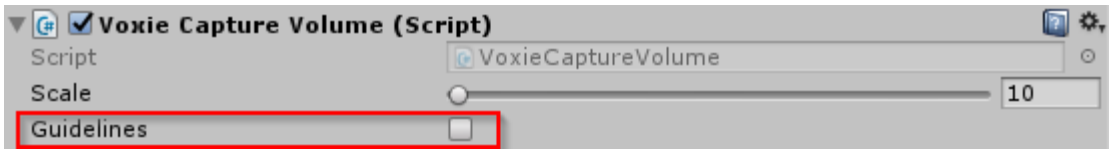


To hide an object which falls within the CaptureVolume set its tag to ‘VoxieHide’.

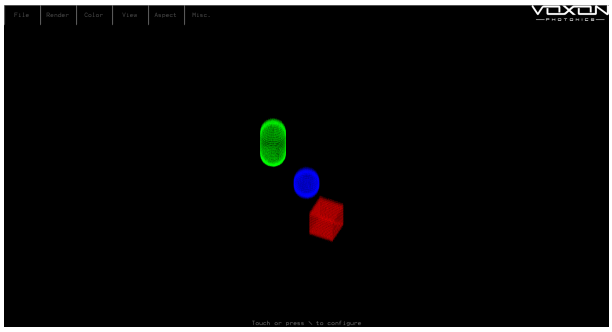


We suggest doing that to any single coloured plane which fills the full volume

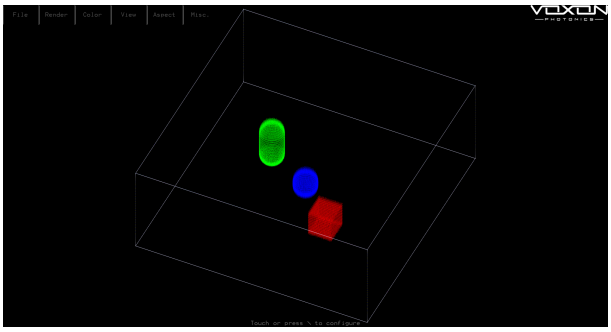
While building your content we suggest using Guidelines; These will clearly illustrate the Rendered area within the VX1’s emulation mode.



WARNING: While active animated meshes within the scene will flicker between animation states. Disable this option before building for VX1



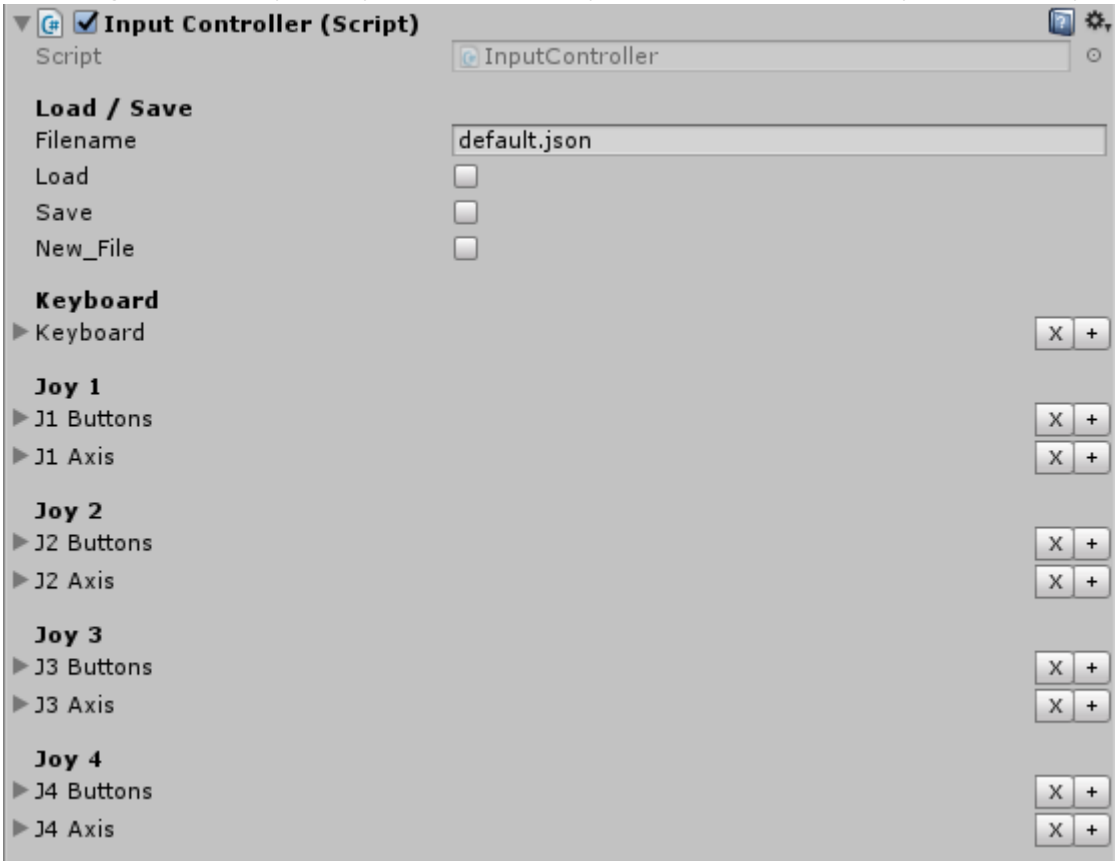
Without Guidelines



With Guidelines

INPUT

Input binding is handled by the Input Controller component attached to the Capture Volume prefab.



Input Scripting

Input is handled by the Voxon.Input class within C# scripts.

Keyboard	GetKey(string: Action) GetKeyDown(string: Action) GetKeyUp(string: Action)
Controller *Default to player 0*	GetButton(string: Action) GetButtonDown(string: Action) GetButtonUp(string: Action) GetAxis(string: Action)

Additionally a set of functions handle multiplayer controllers

Controller	GetButton(string: Action, Int[0-3]: Player) GetButtonDown(string: Action, Int[0-3]: Player) GetButtonUp(string: Action, Int[0-3]: Player) GetAxis(string: Action, Int[0-3]: Player)

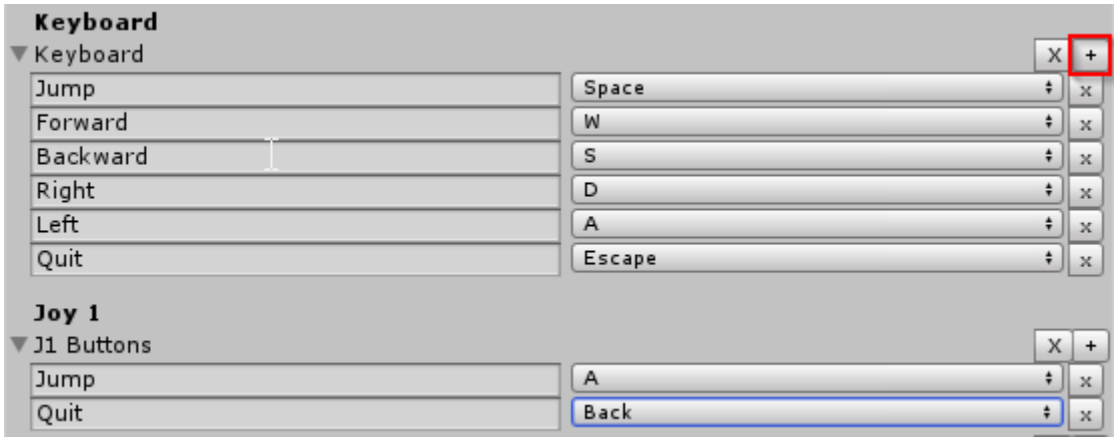
An example movement script is available in VoxonUnity/example (Voxon_Movement.cs).

Note: A default quit system has been included within the CaptureVolume's Update Method, you may wish to remove this once your input system has been implemented.

Setting Bindings

Input is handled via a series of String bindings against a dropdown list of keys. Simple enter the binding name and select the desired key for input.

To add a binding click the + button to the right of the controller

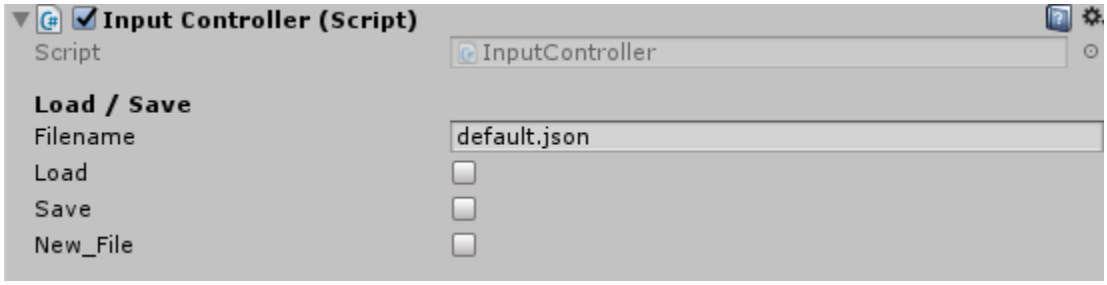


To remove a single binding click the x button next to the binding

To remove all bindings for a controller click the X next to the controller

Saving and Loading

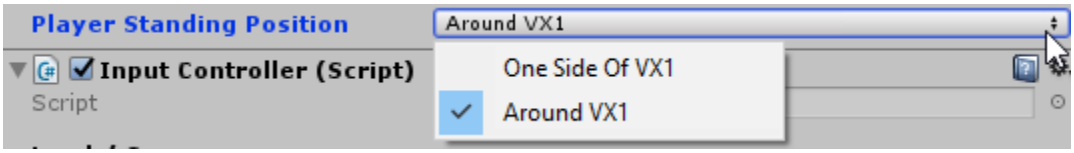
Input Bindings can be saved or loaded on the fly as required. Enter a file name and click the **Load** button to load the file; **Save** will save any bindings to the file and **New_File** will wipe the current bindings.



When filename left empty; controller will default to 'default.json'

Player Positioning

To allow for multiple player positioning the within the Capture Volume component is an option to shift player input based on predicted positioning around the VX1



When set to 'One Side of VX1' all controllers will have the same Axis arrangement.

When set to 'Around VX1', Player 0 and 1 will have inverted controls (as positioned on opposite sides of the device), while player 3 will have their axis swapped, and player 4 having the inverse of player 3's controls.

TESTING

Please note: Due to compatibility issues with the Unity Editor's display window and the VX1, it is strongly recommended testing is performed via Build and Run (CTRL+B) rather than the Play button.

BUILD

When building for the first time go to the **Player Settings** section within the **Build Settings** and ensure they are set at follows.

PLAYER SETTINGS

Before you begin to test ensure that the following player settings are enabled:

Resolution

Default is Full Screen: off

Run in Background: on

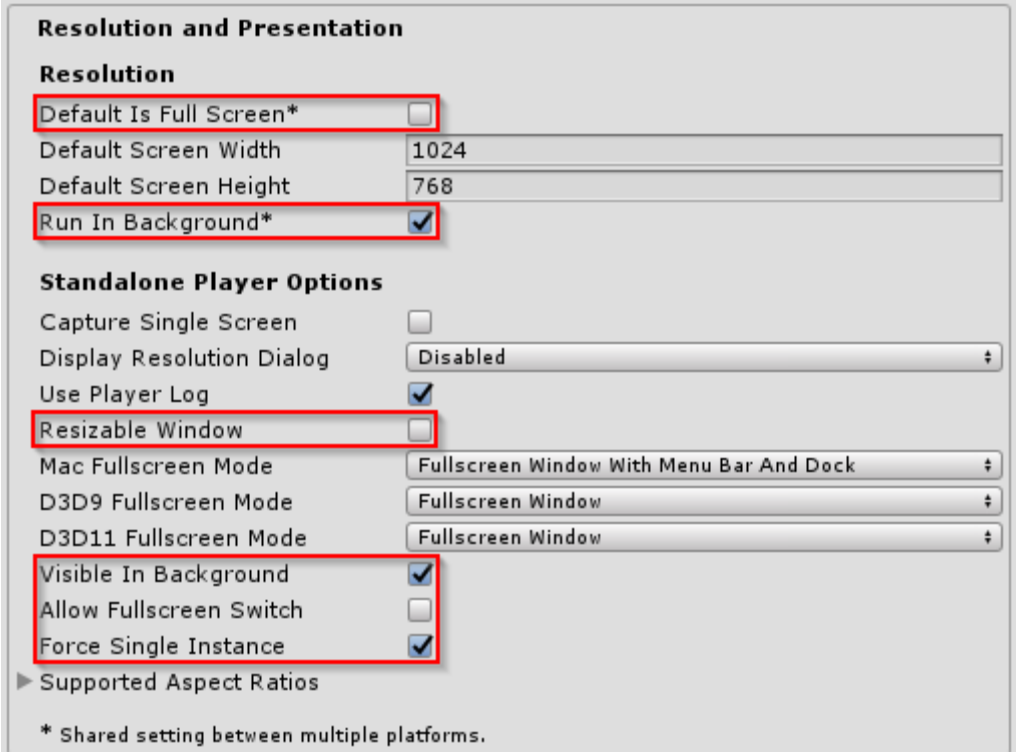
Standalone Player Options

Display Resolution Dialog: Disabled

Visible in Background: on

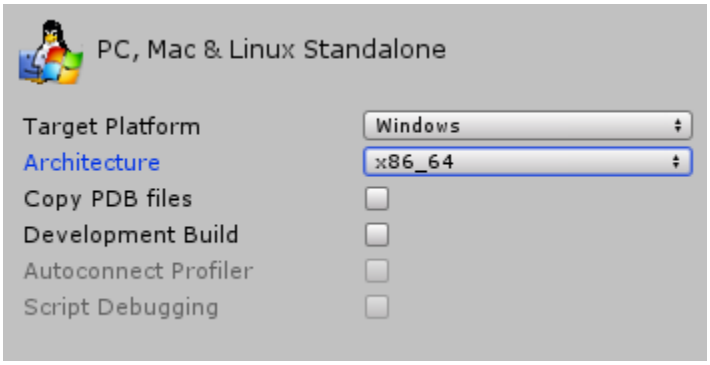
Allow Fullscreen Switch: off

Force Single Instance: on

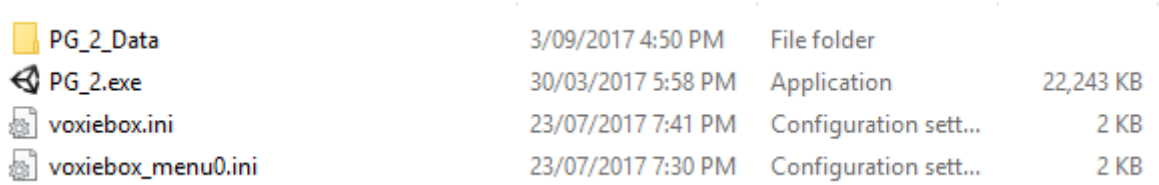


BUILDING

Choose build directory and compile as a **64-bit windows** game.

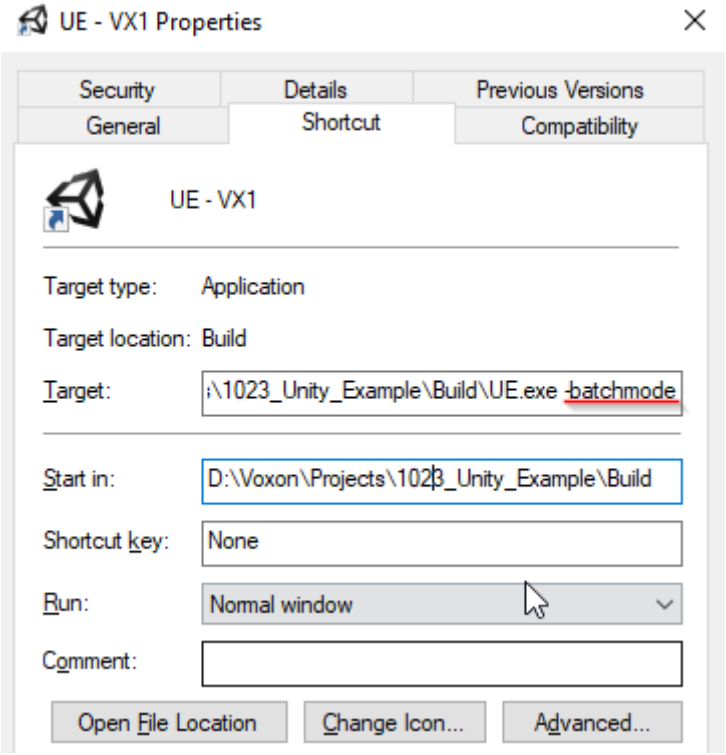


Once compiled add the supplied default voxiebox.ini and voxiebox_menu0.ini to the build directory *Assets/VoxonUnity/default_settings*



These files can then be adapted to meet your program's requirements.

Once built, create a shortcut of the executable with the following flag “-batchmode”



Your game will now only display on the VX1

Following these settings. Your game should now successfully run on the VX1.