a.r.u. | Peterborough

Game Design and Development Workflows

Introduction to

UNREAL ENGINE

YOUR TASKS

TASK ONE: Adding keyboard controls to a game

Using both the guide below and experimentation, add the following controls to the side-scrolling game template provided.

Left Arrow = Move Left
Right Arrow = Move Right
Up Arrow = Jump

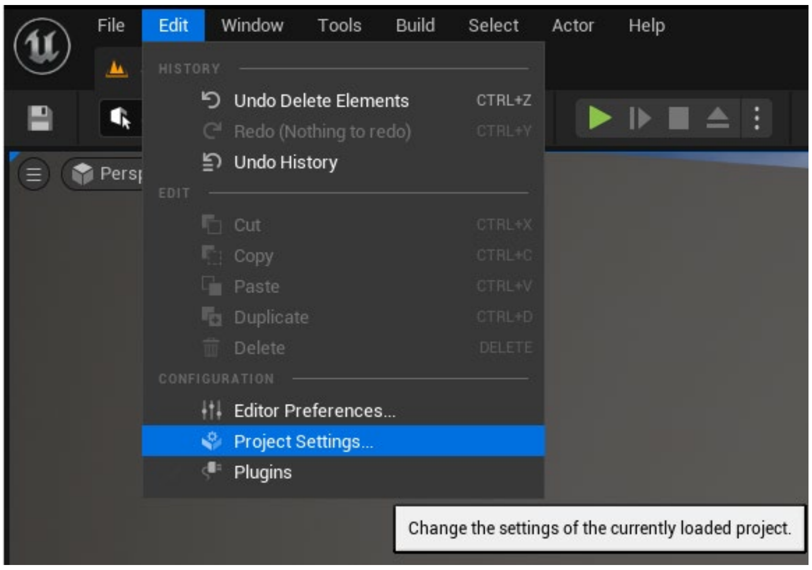
TASK TWO: Explore the level. Can you find the item to collect?

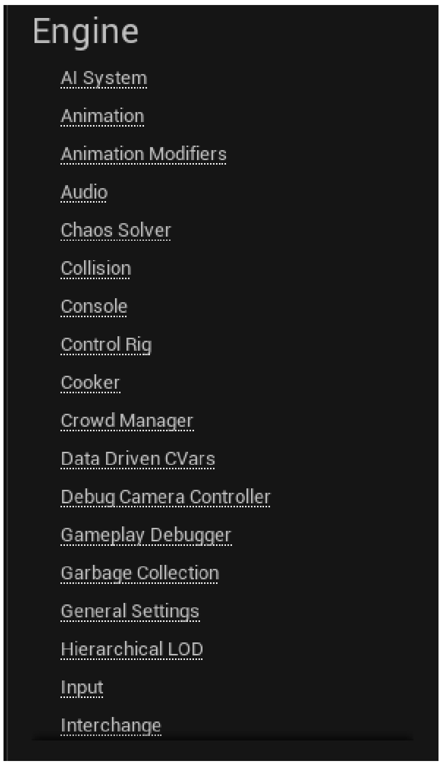
TASK THREE: Extend keyboard control options & further extension activities

TASK ONE:

First, we will take a look at the existing keybindings in the project. Keybindings associate a key, button or gesture with a specific action that can be carried out in-game.

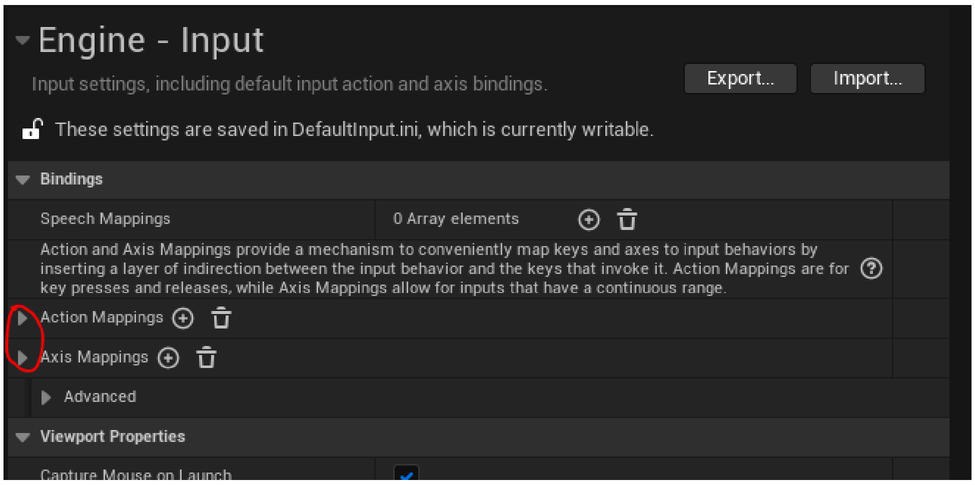
1) From the main UE5 interface, click on **Edit**, then **Project Settings…**





2) In the Project Settings Window, scroll down to **Engine** and select **Input**.

3) From the **Engine – Input > Bindings** menu, expand the **Action Mappings** and **Axis Mappings** options by clicking the small right-facing triangle next to each:



Engine - Input

Input settings, including default input action and axis bindings.

Export... Import...

These settings are saved in DefaultInput.ini, which is currently writable.

Bindings

Speech Mappings 0 Array elements

Action and Axis Mappings provide a mechanism to conveniently map keys and axes to input behaviors by inserting a layer of indirection between the input behavior and the keys that invoke it. Action Mappings are for key presses and releases, while Axis Mappings allow for inputs that have a continuous range.

Action Mappings

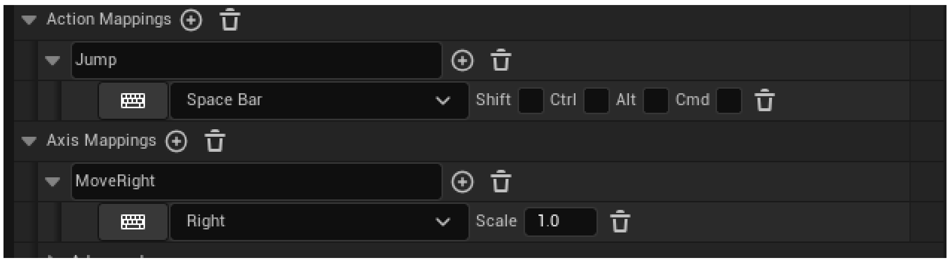
Axis Mappings

Advanced

Viewport Properties

Capture Mouse on Launch

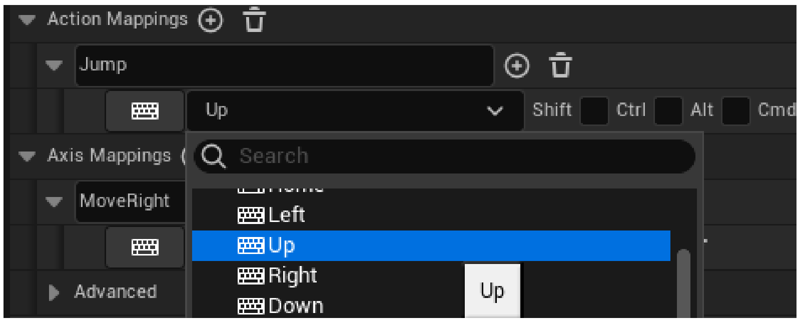
4) Here you will see the pre-configured Jump and MoveRight actions. Expand these also to see which keys they are mapped to.



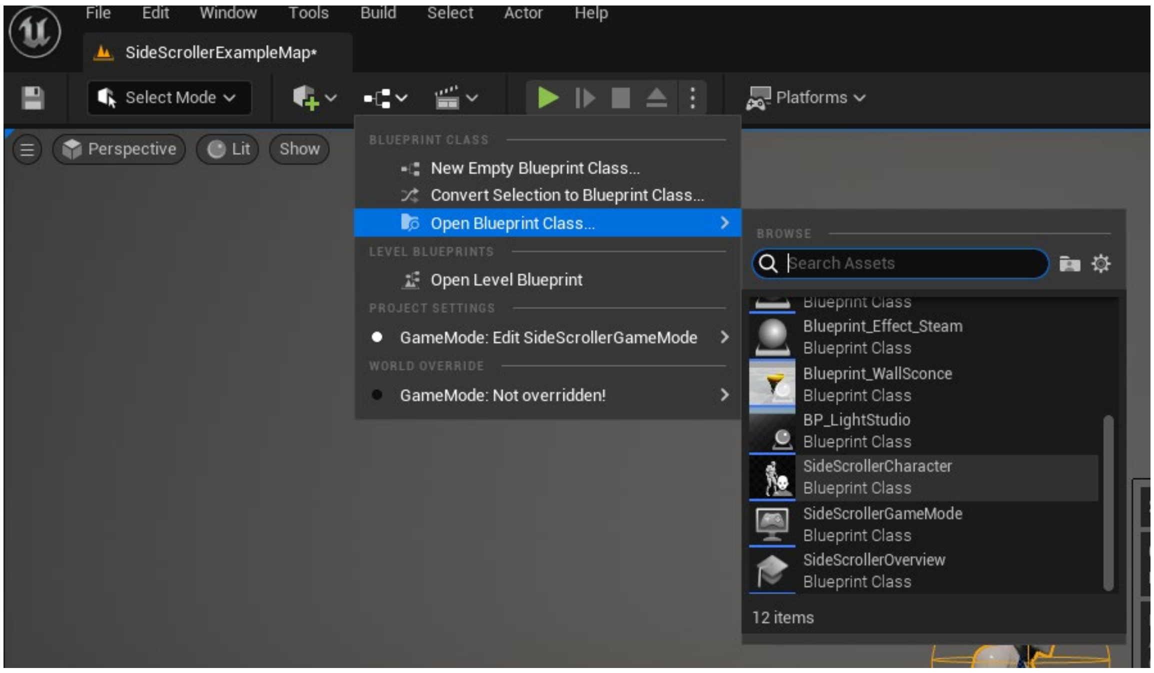
#

5) You can see that the **Jump** action is mapped to Space Bar and the **MoveRight** action is mapped to the Right arrow key. Note that the **Right** keypress has a scale of 1.0; this is important for later.

6) Change the **Jump** action from **Space Bar** to the **Up** arrow. You can do this by either clicking on the keyboard icon and pressing the key on the keyboard, or by selecting the Up key from the drop-down menu.

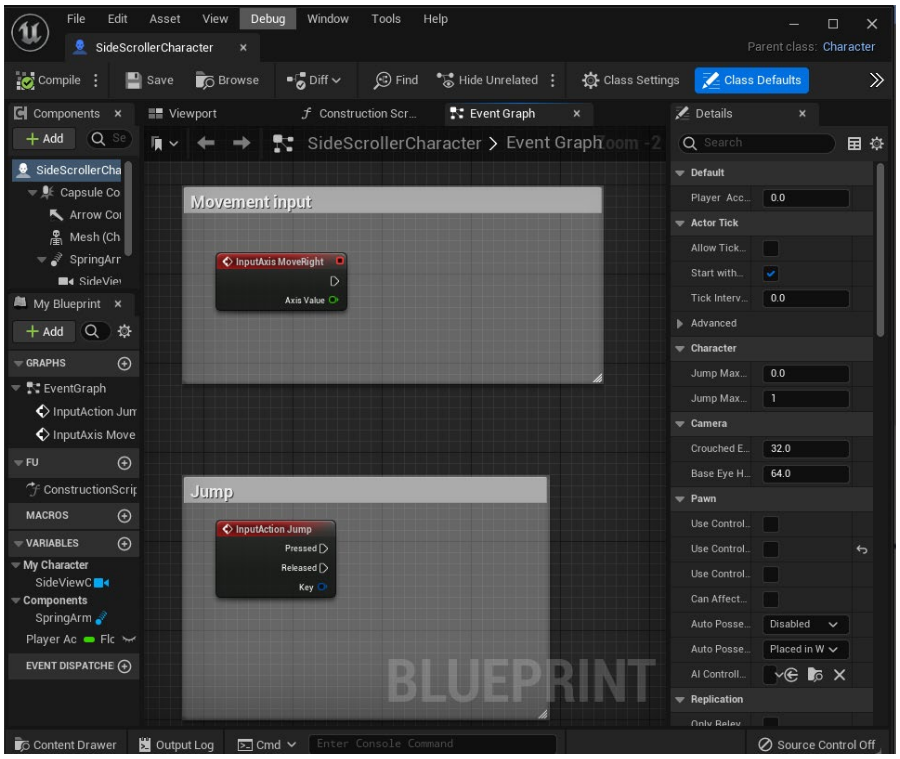


6) Now we know how keys are mapped, close the **Project Settings** window and click on the Blueprints icon from the UE5 top taskbar, and choose **Open Blueprint Class > SideScrollerCharacter**

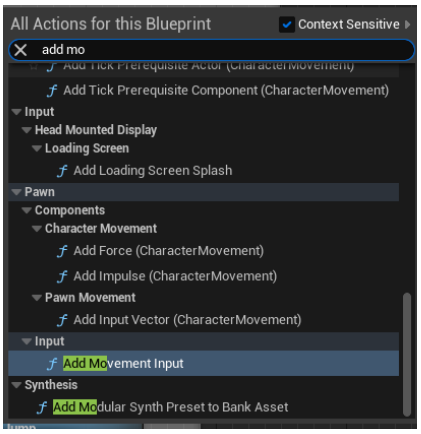


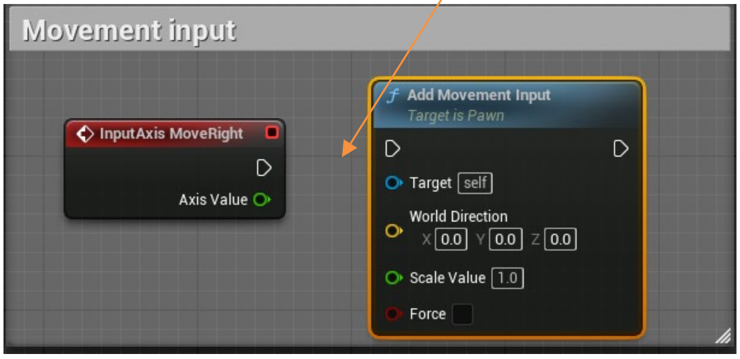
A Blueprint is a visual scripting tool that enables you to program functionality quickly into your games without having to use code. However, Unreal Engine also includes the ability to fully code your games using the C++ language if you would like to have more control and the ability to optimise routines. Games made in the industry typically use a combination of both Blueprints and Code in the development process.

7) You will now see the Blueprint '**Event Graph'** editor, with two Input events already added; **InutAxis MoveRight** and **InputAction Jump**.

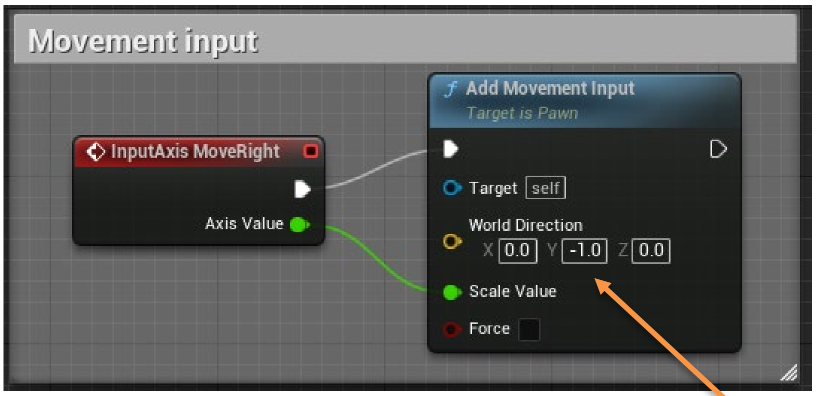


8) To enable movement of our SideScollerCharacter 'pawn' we need to tell UE5 what to do if the MoveRight action is triggered (i.e. when the key bound to this action is pressed). Add an **Add Movement Input** node to the **Movement input** graph by right-clicking near to **InputAxis MoveRight** and searching for **Add Movement Input** in the menu that appears.


All Actions for this Blueprint
Context Sensitive
add mo
Add Tick Prerequisite Component (CharacterMovement)
Input
Head Mounted Display
Loading Screen
Add Loading Screen Splash
Pawn
Components
Character Movement
Add Force (CharacterMovement)
Add Impulse (CharacterMovement)
Pawn Movement
Add Input Vector (CharacterMovement)
Input
Add Movement Input
Synthesis
Add Modular Synth Preset to Bank Asset



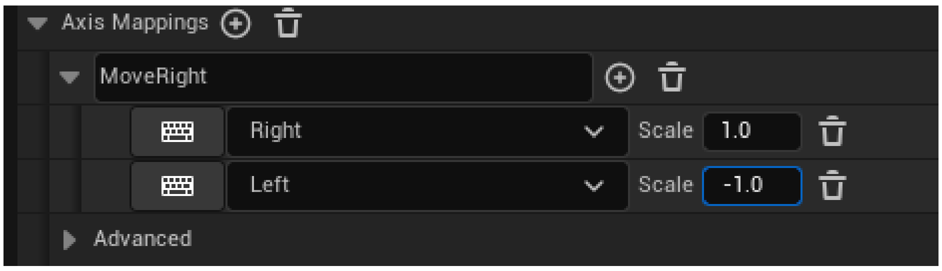
9) Now connect the MoveRight action to the Movement Input node by dragging from the White right-triangle output to the matching input. Then do the same for the **Axis Value** by linking it to **Scale Value**.



10) Finally, change the World Direction Y value from 0.0 to -1.0. This will tell the 'pawn', which is the SideScrollerCharacter to move in the 'negative Y axis', which in our screen-space moves the player to the right.

11) Test the game by pressing the green Play button on the top taskbar. Make sure the gameplay preview window is active by clicking on the level editor near the player, and then press the Right key to test player movement. Notice that you cannot currently move left. Once you are done, press the Escape key to regain mouse control.

12) Now that you have the player moving to the right we need to get them to move left. Navigate back to the **Engine – Input > Bindings** menu in **Project Settings…** and add a second Mapping to the MoveRight action by clicking the ⊕ icon next to it. Set this mapping to Left and the Scale to -1.0

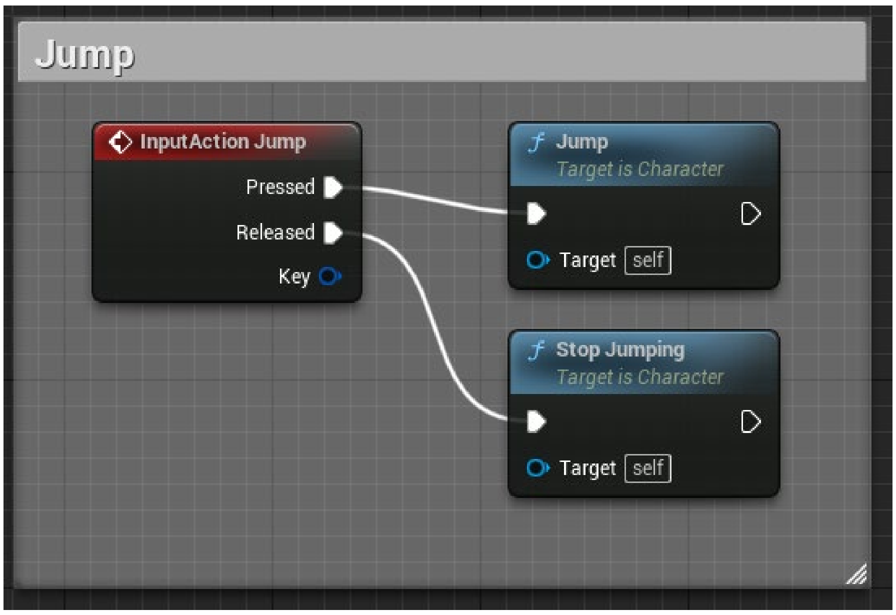


Right Scale 1.0
Left Scale -1.0

The Scale value is what is passed into the Add Movement Input node (See step 9) and by setting the scale to -1.0, we invert the value in World Direction Y (i.e. this switches the direction from -1.0 to 1.0 and makes the player move left)

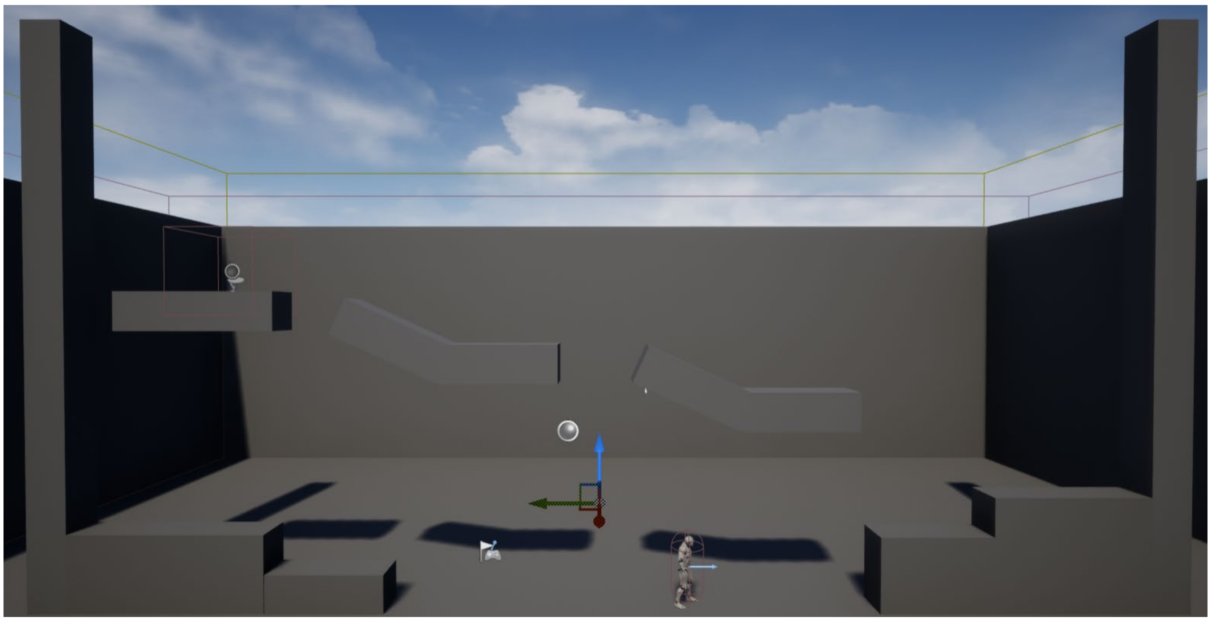
13) Now close the **Project Settings** window and play your game again to test it. You should now be able to move both right and left!

14) Try adding the following events to the Jump graph in the **SideScollerCharacter** blueprint.



<u>TASK TWO:</u>
Now when you test your game you should be able to jump by pressing the Up arrow. Try exploring the level, and if you find the statue, press the **E key** to collect it; If you want to see where this is scripted, check out the **InteractableItemBP** blueprint.

Well done, you now have a simple working game!



<u>TASK THREE:</u> Extension challenges

a) Replicating the techniques from task one, add additional keyboard control options.

A = Move Left
D = Move Right
W = Jump

b) Can you change the text that is displayed on the screen when you collect the statue?
c) Can you add some more platforms to the level? You will need to add a cube from the **Quickly add to project** button and selecting **Shapes** > **Cube**.
   Can you set the properties of the new cube to match the other platforms (Location, Material, etc).
   Ensure that you add the new platform to the Ledges folder in the Outliner and name it to match the other items.