# Workshop 8 Lab 1

In this activity, we are going to first use the old Input Manager to control the Player using keyboard, then we install the new Input System to control the Player using keyboard.

**Part 1: Input Manager**

**Procedure:**

New a Unity3D project, name it PlayerControl.

Insert a plane, scale XYZ by 10.

Graphical user interface

Description automatically generated with medium confidence

Insert a cube, name it as Player, place the cube on the plane, Y set to 0.5.

Graphical user interface, application

Description automatically generated

Attached a RigidBody to the Player.

Graphical user interface

Description automatically generated

Set Drag and Angular Drag to 0.5

Freeze Rotation X and Z

Create a C# script OldPlayerControler

Attached the script to Player

Text

Description automatically generated

Test the scene.

Use up and down arrow key to move the player forward and backward.

Use left and right arrow key to turn the player left and right.

**Part 2: Input System**

**Procedure:**

Add in a new Input System

Windows->Package Manager

Change to Unity Registry

Graphical user interface, text, application, chat or text message

Description automatically generated

Look for Input System and Install.

Graphical user interface, text, application

Description automatically generated

This will restart Unity3D.

Note that once this is installed, we will not be able to run the scene because the C# script is not compatible to the new Input system.

To change back to the old Input system, got to

Build Settings->Player Settings->Player->Other Settings->Configuration->Active Input Handling\* to change back to Input Manager (old)

Graphical user interface, text, application, chat or text message

Description automatically generated

Remove the OldPlayerMovement component from Player

Add in Player Input component

Graphical user interface, text, application, email

Description automatically generated

Now we need to create an Input Action Asset for Player Input Component.

Create a new folder “Input Actions”

In Input Actions folder, create a new Input Actions (Create->Input Actions) and name it Player Control.

Double click on Player Control to open up the window.

Graphical user interface, application, Word

Description automatically generated

Note: In this exercise we are just configuring the keyboard input.

The Input system allows us to configure many other input devices like Gamepads.

There are three columns in the Windows

Action Maps -> Group of Controls (i.e. for different type of players and situations)

Actions -> actions perform in the scene, methods in a class in this case.

Properties -> Properties of the actions

New an Action Maps, name it as Gameplay

Table

Description automatically generated with medium confidence

Under Gameplay, rename the New Action to Move.

Graphical user interface, text, application, chat or text message

Description automatically generated

Configure the Move properties.

Graphical user interface, application

Description automatically generated

We want to read in numbers (value) and we want to read in X and Y value (Vector2)

Under each Action, we need to bind the action to a physical input (e.g. a button on a keyboard or a stick on a gamepad.

Now we want to bind the Move Action. (Click on the arrow besides Move)

Application

Description automatically generated with low confidence

Delete <No Binding>

Add in a Up/Down/Left/Right Composite (2D Vector)

Rename it as KeyboardGraphical user interface, text, application, chat or text message

Description automatically generated

Map the Path of Up to Up Arrow[Keyoard]

Keyboard->By Location of Key(Using US Layout)->Up Arrow(Current Layout:Up)

Graphical user interface, application

Description automatically generated

Map Down, Left and Right using the same method.

Text

Description automatically generated with medium confidence

Add in one more Action “Jump”

A picture containing graphical user interface

Description automatically generated

Configure No Binding and bind it to Keyboard Spacebar.

Graphical user interface, application, table

Description automatically generated

Click on Save Assets on the PlayerControl window.

Graphical user interface, application

Description automatically generated

Make sure the \* is gone after saving.

Graphical user interface, text, application

Description automatically generated

Drag and drop PlayerControl onto Player game object Player Input

Graphical user interface, text, application

Description automatically generated

New a NewPlayerControl C# script.

Text, application

Description automatically generated

Attached the script to Player.

Test the scene.

Use up and down arrow key to move the player forward and backward.

Use left and right arrow key to turn the player left and right.