THANKS FOR BUYING HORROR FPS KIT!

If you like our assets, you can also visit our channel:

www.youtube.com/c/ThunderWireGamesIndie

and check out our tutorials and game developments.

Also, you can check out our website for future asset releases www.twgamesdev.com

ABOUT HORROR FPS KIT

HORROR FPS KIT is an advanced and easy-to-use horror game template with many features essential to creating your own **horror** game, including gameplay features seen in **AAA** horror games of the last decade. It contains a lot of ready-to-use assets, just drag and drop into a scene.

SUMMARY

FEATURES

PLAYER FUNCTIONS

- Player Controller (Walk, Run, Jump, Crouch, Prone, Ladder Climbing)
- Fully Animated Player Body (Movement)
- Texture and Tag based Footsteps
- Player Lean and Wall Detection
- Weapons (Pistol, Shotgun, Axe)
- Zoom Effect
- Item Switcher

SYSTEMS

- Save/Load System (Player Data, Scene Data, Inventory, Encryption)
- Cross-Platform Input (PC, PS4, XONE)
- Advanced Menu System (Cross-Platform Support)
- Inventory System (Add, Remove, Move, Use, Drop, Examine, Store, Shortcuts)
- Objectives System (Add, Complete, Complete and Add, Events)
- Cutscenes System (Queue)
- Examination System (Rotate, Double Examine, Papers)
- Drag Rigidbody System (Rotate, Zoom, Throw)
- Interact Manager (Pickup, Messages)
- Floating Icons System

OBJECT PICKUPS

- Custom Objects Pickup
- Light Items (Flashlight, Hinge Lantern, Candle)
- Backpack Inventory Expand Pickup

DYNAMIC OBJECTS

- Door, Drawer, Lever, Valve, Movable Interact
- Types (Mouse, Animation, Locked, Jammed)
- More Objects (Keypad Lock, Keycard Lock, Padlock)

MORE FEATURES

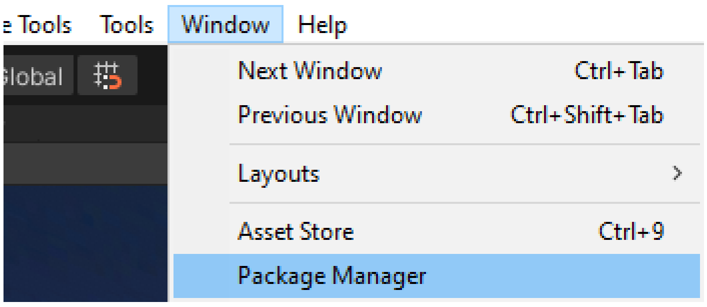
- Rebindable Keyboard Controls at Runtime
- Zombie AI (Sleep, Wander, Scream, Agony, Hunger, Attack, Attract, Footsteps)
- Scene Preloader (Background Change, Tips)
- Jumpscares (Animation, Scare Effects, Scared Breath, Customizable)
- UI Menu (Main Menu, Pause Menu, Load Menu..)
- Notifications (Pickup, Hint, Messages)
- Realistic VHS Player
- CCTV System
- Ambience Zones
- Water Buoyancy
- Interactive Lights (Lamps, Switchers, Animation)
- Props, Collectable Items, much more..

MANUAL PROJECT SETUP

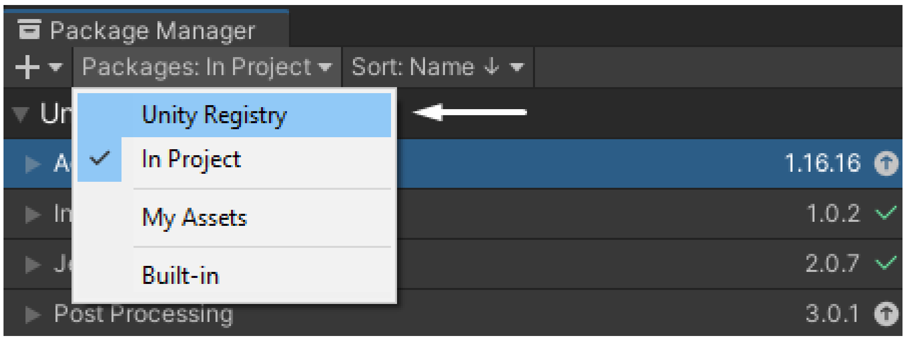
We strongly recommend importing asset into an empty project and importing alltags and layers!

Make sure that you have imported **Input System**, **Post Processing** and **Timeline** from **Package Manager!**

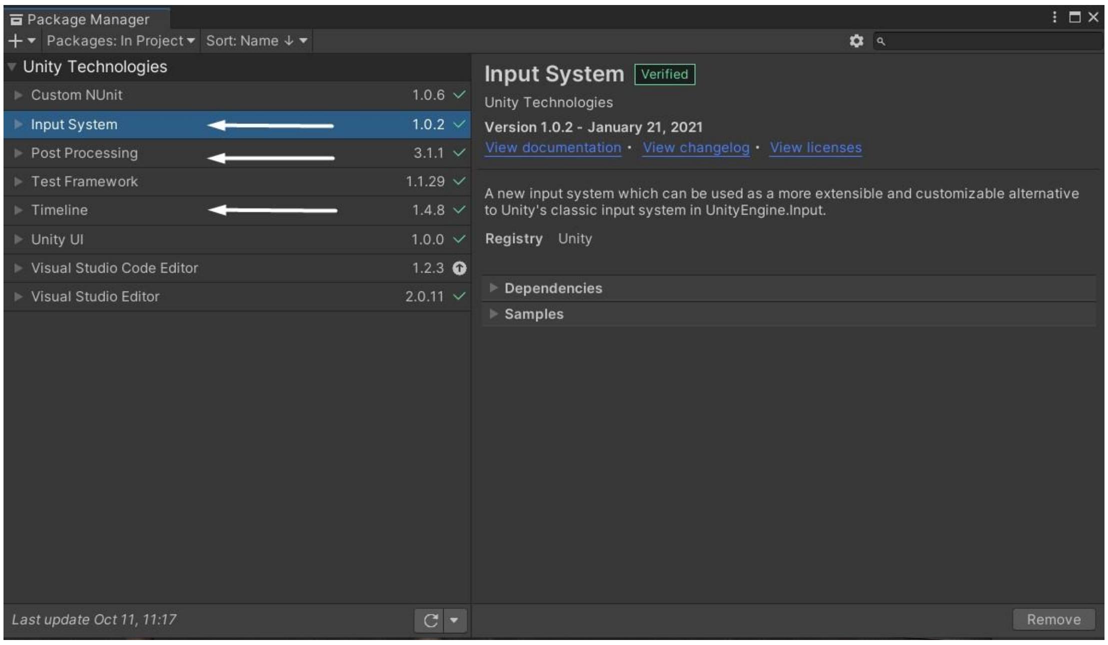
1. Open **Package Manager** from **Window** drop-down.



2. Select **Unity Registry** from the **Packages** drop-down menu.



3. Find the packages mentioned above and import them into a project.



Custom NUnit	1.0.6
Input System	1.0.2
Post Processing	3.1.1
Test Framework	1.1.29
Timeline	1.4.8
Unity UI	1.0.0
Visual Studio Code Editor	1.2.3
Visual Studio Editor	2.0.11

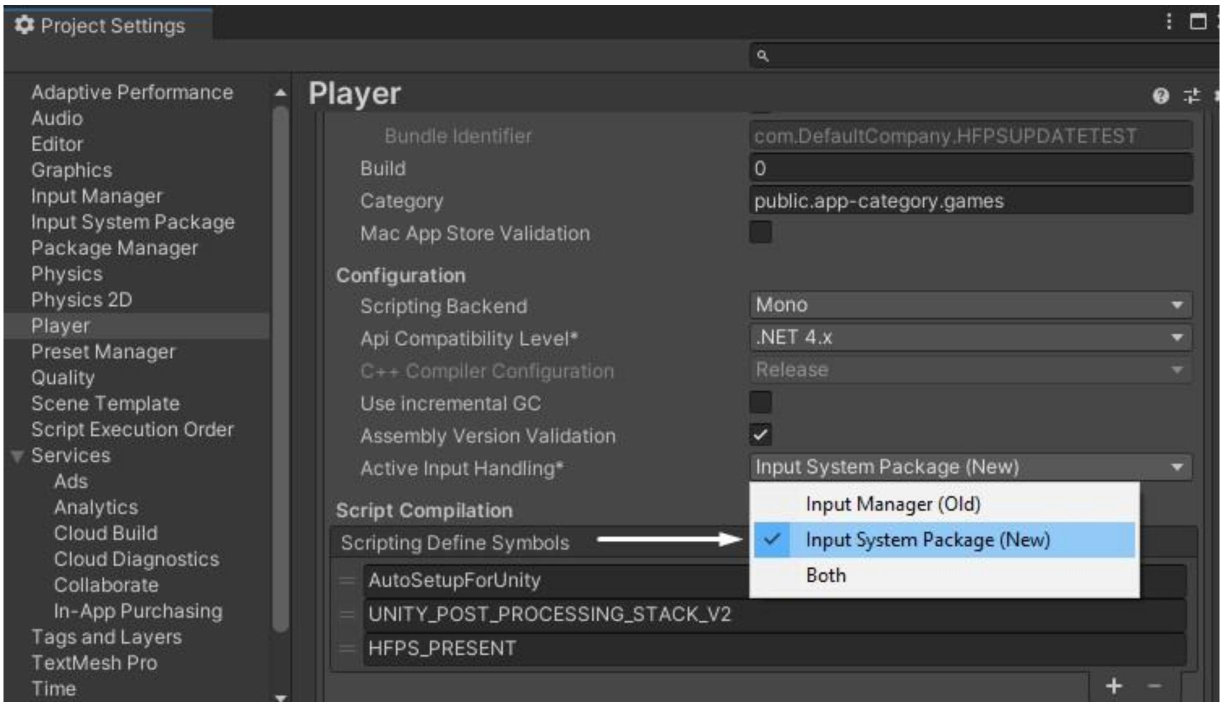
Input System Verified
Unity Technologies
Version 1.0.2 - January 21, 2021
View documentation • View changelog • View licenses

A new input system which can be used as a more extensible and customizable alternative to Unity's classic input system in UnityEngine.Input.

Registry Unity

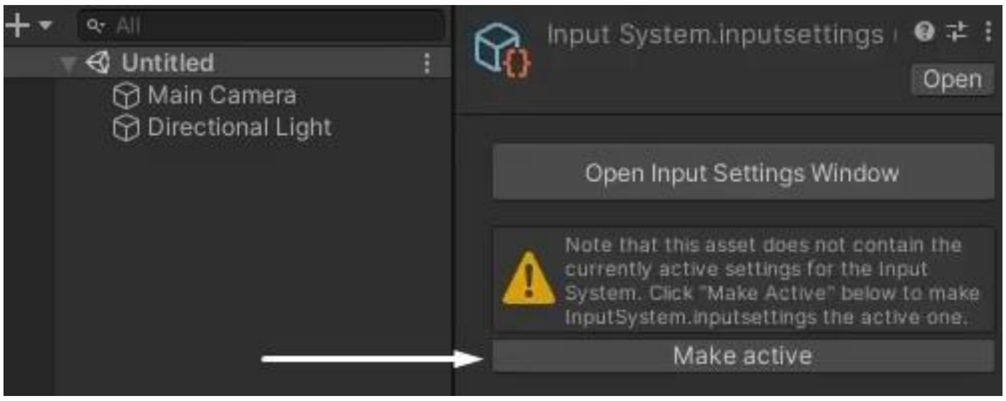
Dependencies
Samples

4. Go to **Player Settings** and set **Active Input Handling** to **Input System Package**.



Bundle Identifier: com.DefaultCompany.HFPSUPDATETEST
Build: 0
Category: public.app-category.games
Mac App Store Validation
Scripting Backend: Mono
Api Compatibility Level*: .NET 4.x
C++ Compiler Configuration: Release
Use incremental GC
Assembly Version Validation ✓
Active Input Handling*: Input System Package (New)
Scripting Define Symbols
AutoSetupForUnity
UNITY_POST_PROCESSING_STACK_V2
HFPS_PRESENT

5. Go to **Horror FPS KIT\HFPS Assets\Scriptables\InputSystem** folder and click activate in **InputSystem.inputsettings**.



Note that this asset does not contain the currently active settings for the Input System. Click "Make Active" below to make InputSystem.inputsettings the active one.

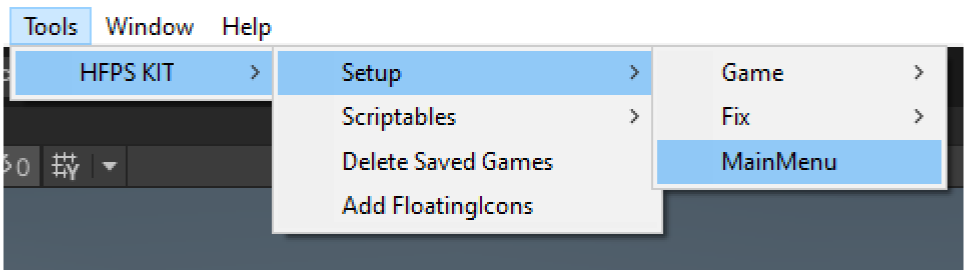
- If this button does not exist, ignore this step.

6. If you see this message after clicking the Play button - **"[Device Init] Supported Input Device is not connected!"**, or the game does not respond to any inputs just restart the Unity.

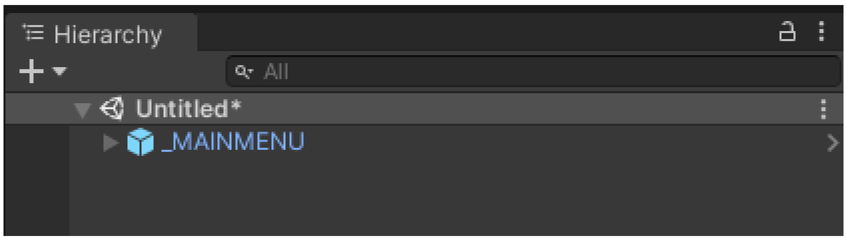
- If you have trouble setting up a project along with HFPS, here is a video on how to set up a project properly: https://www.youtube.com/watch?v=iX5voeju3MQ

MAIN MENU SETUP

- Open a new empty scene.
- Go to **Tools -> HFPS KIT -> Setup -> MainMenu**.



- You should now see the **_MAINMENU** object added to your scene.

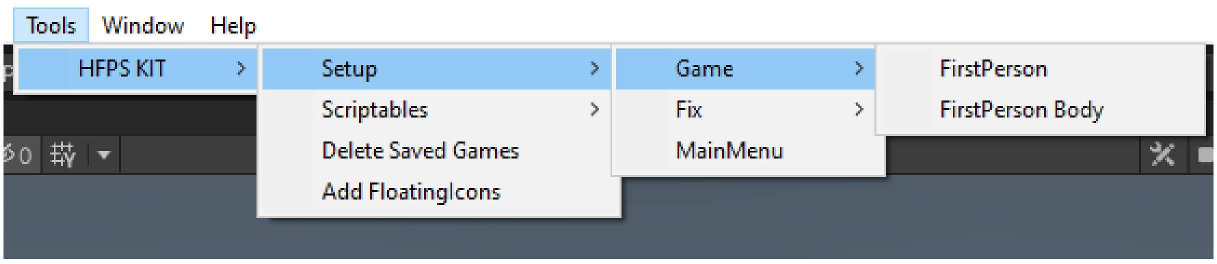


- Add a Camera object to your scene.

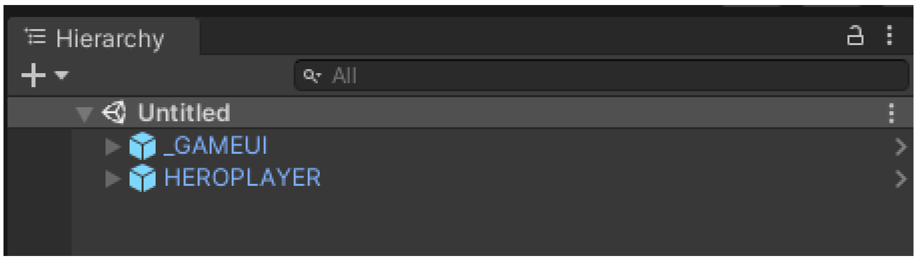
GAME SETUP

The steps are the same as when setting up the main menu.

1. Open a new empty scene.
2. Go to **Tools -> HFPS KIT -> Setup -> Game**.

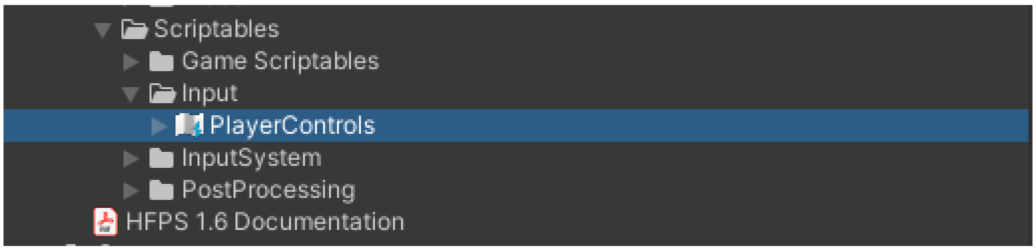


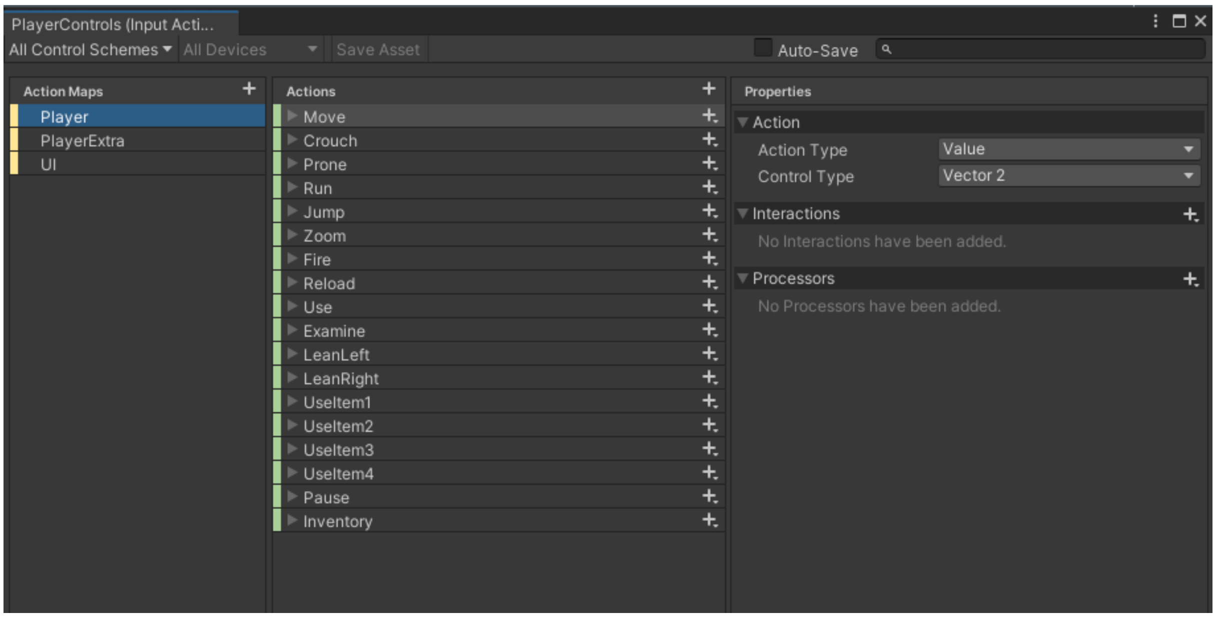
3. You should now see the **_GAMEUI** and **HEROPLAYER/FPSPLAYER** objects added to your scene.



ADDING NEW INPUTS

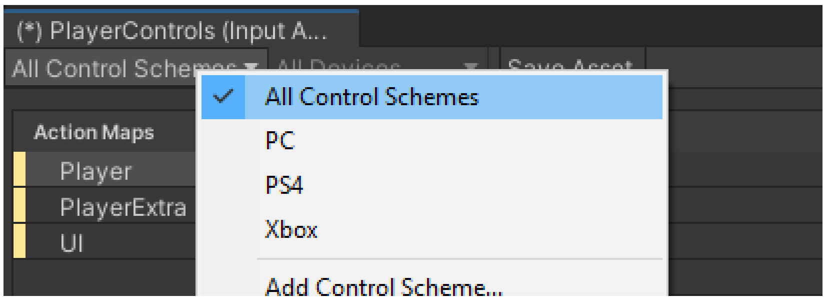
1. Open the **PlayerControls** asset window by double-clicking on it.



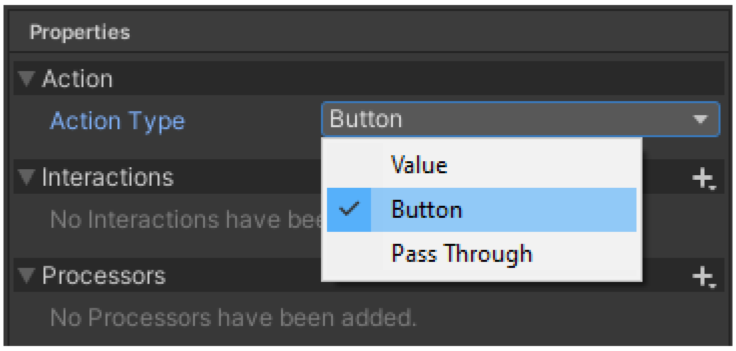


Action Type: Value
Control Type: Vector 2

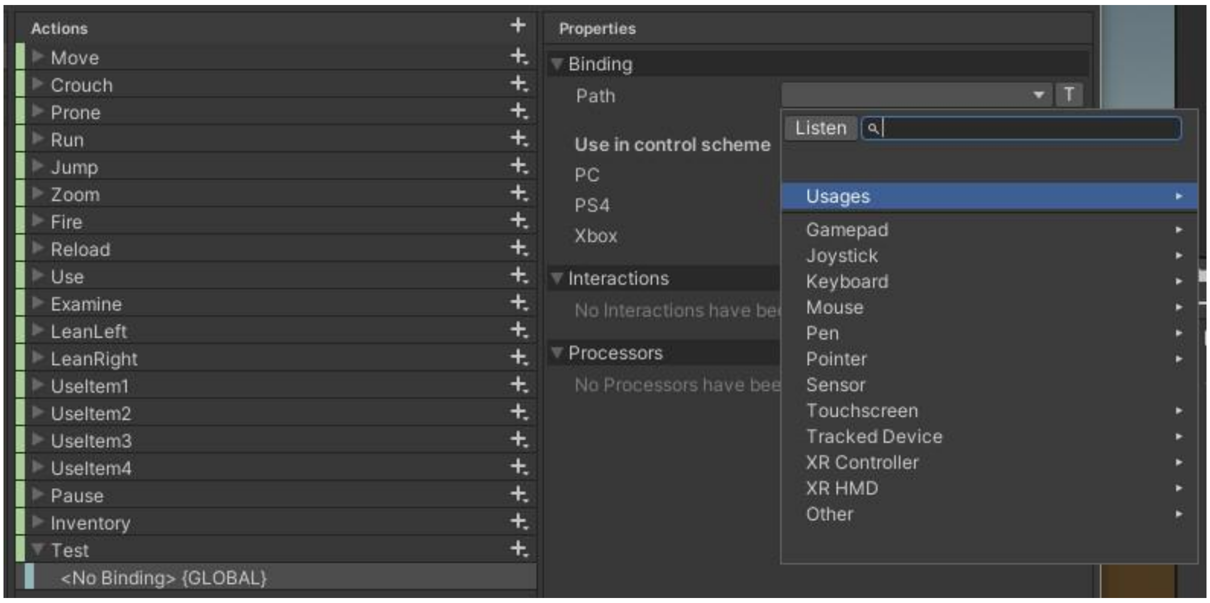
2. Clicking on the **+** button you will add a new action.
3. In the top panel select the control scheme.



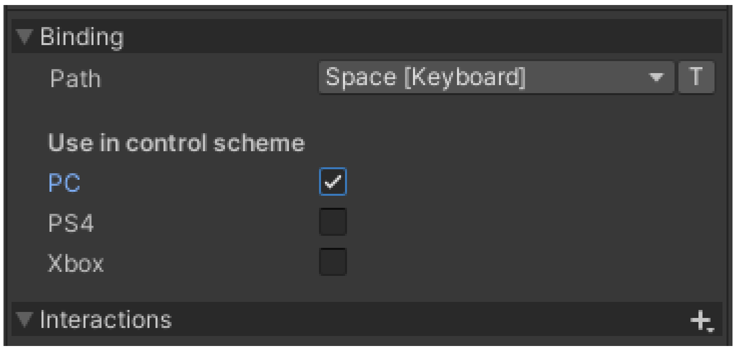
4. After clicking on the action, you can select the **Action Type**.



5. Now select the binding path. The binding path will vary depending on the currently selected action type!



6. After selecting the binding path, select the control scheme that belongs to the input.



Binding

Path: Space [Keyboard]

Use in control scheme

PC ✓

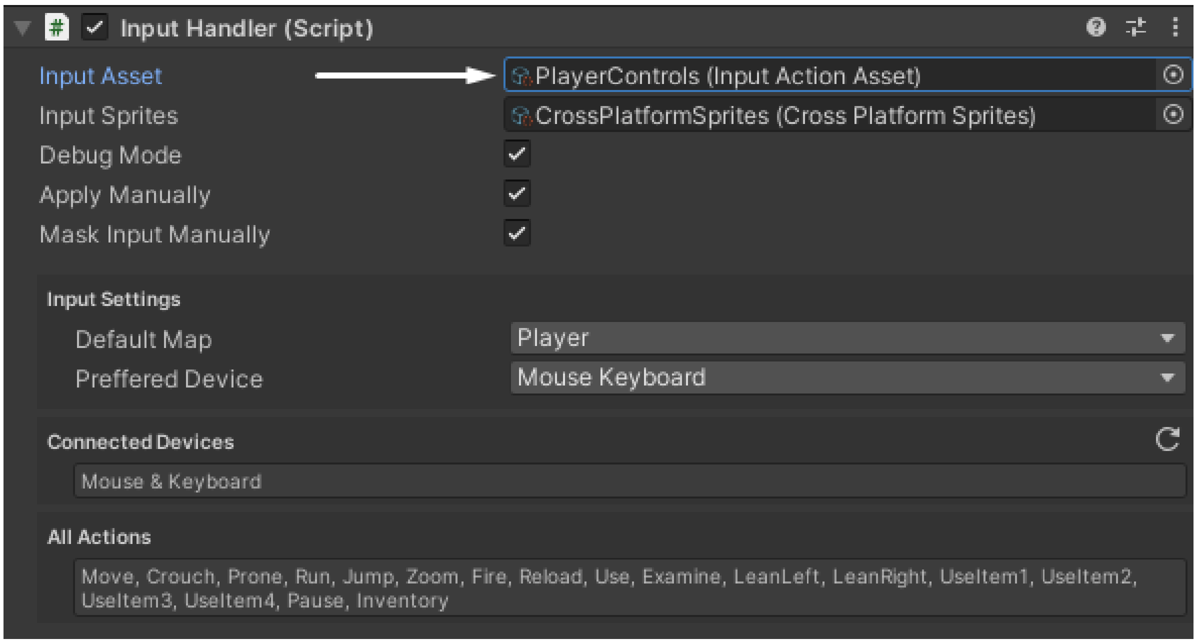
PS4

Xbox

Interactions

7. Now click the **Save Asset** button to save all input changes.

- If you create a new **Input Action Asset**, you must assign it to an **Input Handler** to use it.



Input Handler (Script)

Input Asset	PlayerControls (Input Action Asset)
Input Sprites	CrossPlatformSprites (Cross Platform Sprites)
Debug Mode	✓
Apply Manually	✓
Mask Input Manually	✓

Input Settings

Default Map	Player
Preffered Device	Mouse Keyboard

Connected Devices

Mouse & Keyboard

All Actions

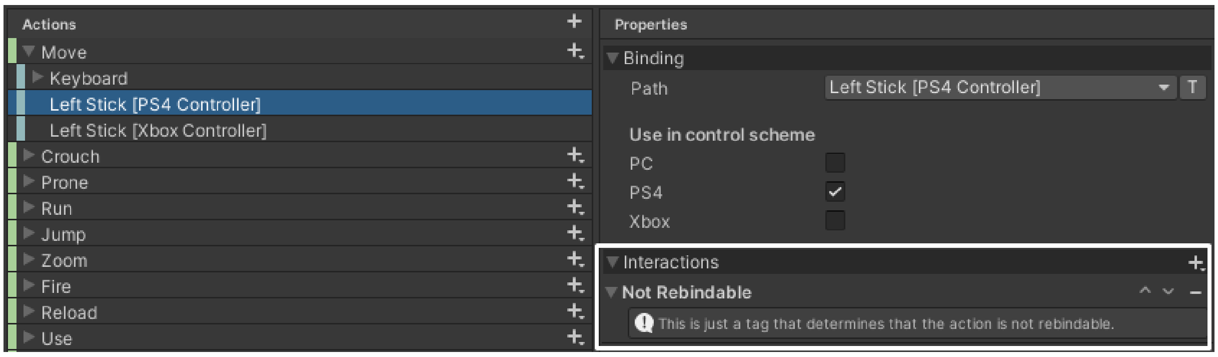
Move, Crouch, Prone, Run, Jump, Zoom, Fire, Reload, Use, Examine, LeanLeft, LeanRight, UseItem1, UseItem2, UseItem3, UseItem4, Pause, Inventory

8. After adding a new action, you can use it using the **Input Handler** functions.

```
if (InputHandler.ReadButtonOnce(this, ActionName))
{
    Debug.Log($"Action {ActionName} is pressed once!");
}
```

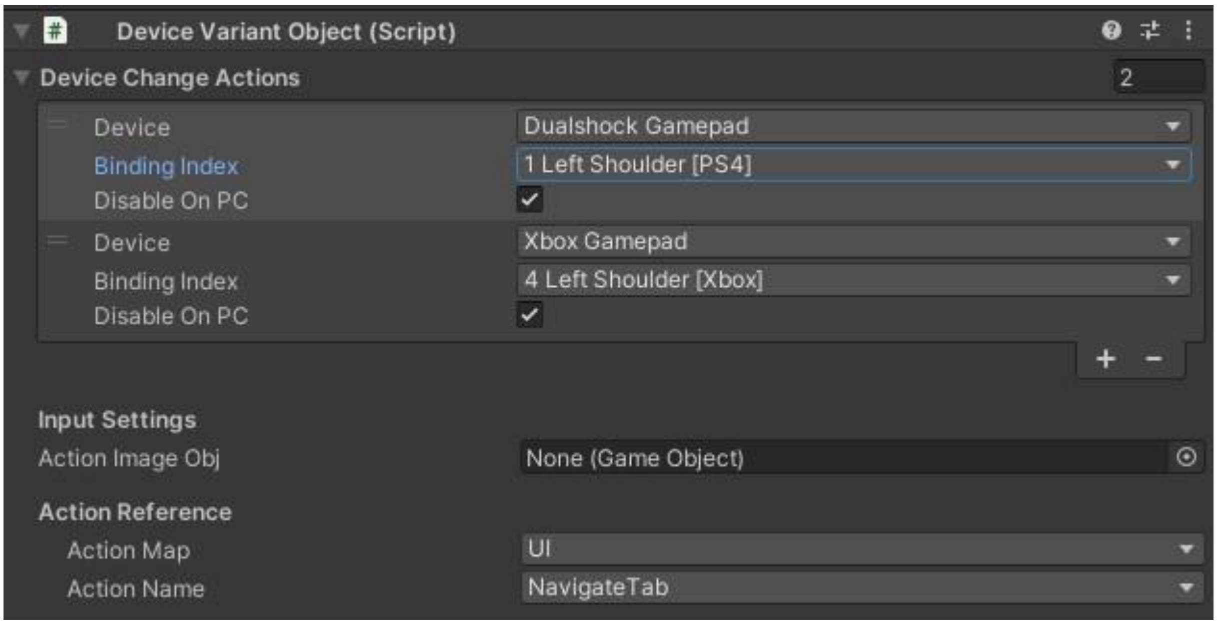
```
Vector2 movement;
if ((movement = InputHandler.ReadInput<Vector2>(ActionName)) != null)
{
    Debug.Log($"Action {ActionName} is updating {movement}!");
}
```

- When using the **ReadInput<T>** function, pay attention to using the return type that belongs to the action specified by the **Action Type**.
- You can specify which inputs cannot be rebinded by adding a **Not Rebindable** interaction.


Actions
Move
Keyboard
Left Stick [PS4 Controller]
Left Stick [Xbox Controller]
Crouch
Prone
Run
Jump
Zoom
Fire
Reload
Use

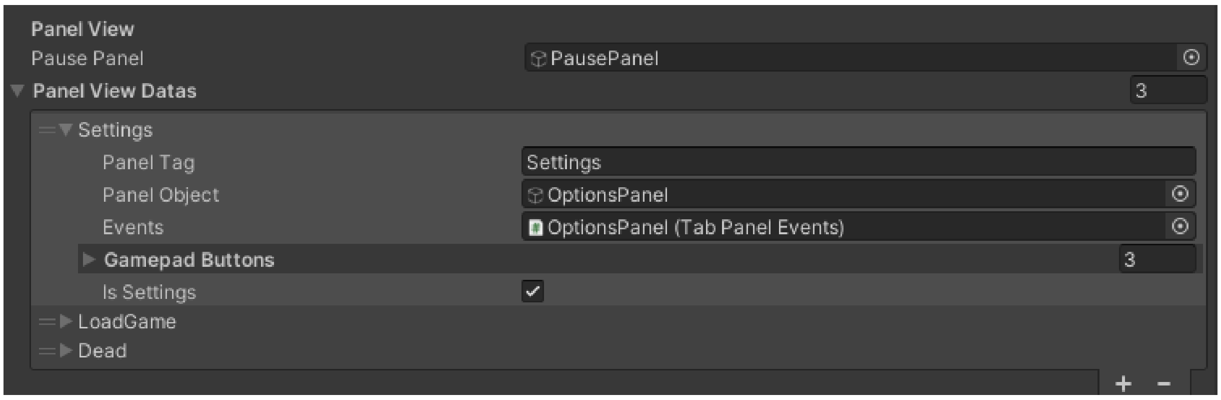
Properties
Binding
Path: Left Stick [PS4 Controller]
Use in control scheme
PC: ☐
PS4: ✓
Xbox: ☐
Interactions
Not Rebindable
This is just a tag that determines that the action is not rebindable.

- If you have problems adding new inputs, you can see the official tutorial for adding new inputs from Unity: https://www.youtube.com/watch?v=5tOOstXaIKE

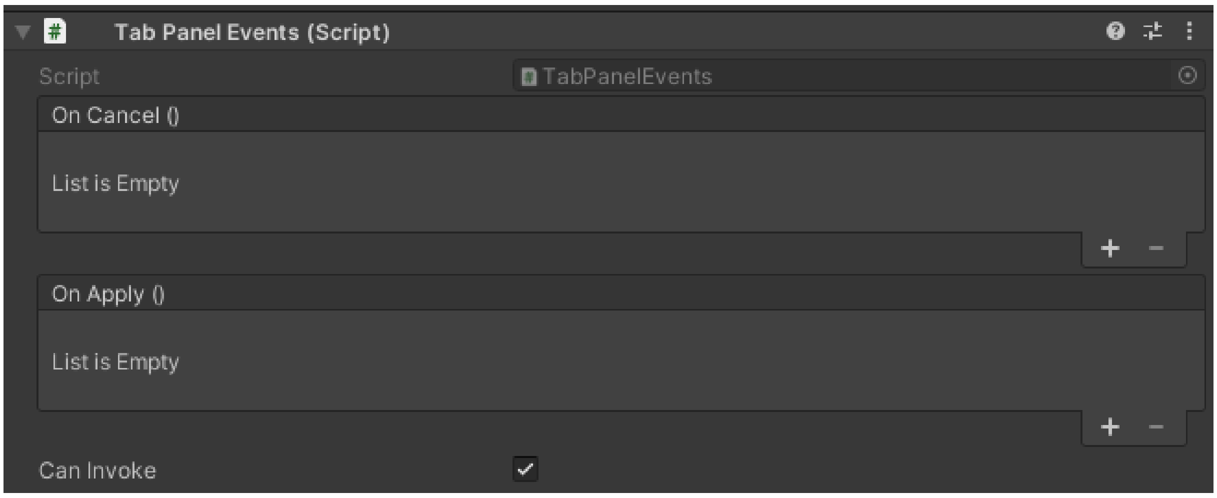

Device Variant Object (Script)
Device Change Actions: 2
Device: Dualshock Gamepad
Binding Index: 1 Left Shoulder [PS4]
Disable On PC: ✓
Device: Xbox Gamepad
Binding Index: 4 Left Shoulder [Xbox]
Disable On PC: ✓
Input Settings
Action Image Obj: None (Game Object)
Action Reference
Action Map: UI
Action Name: NavigateTab

- You can define input icon using the **Device Variant Object** script, which changes the icon depending on the currently selected input controller.

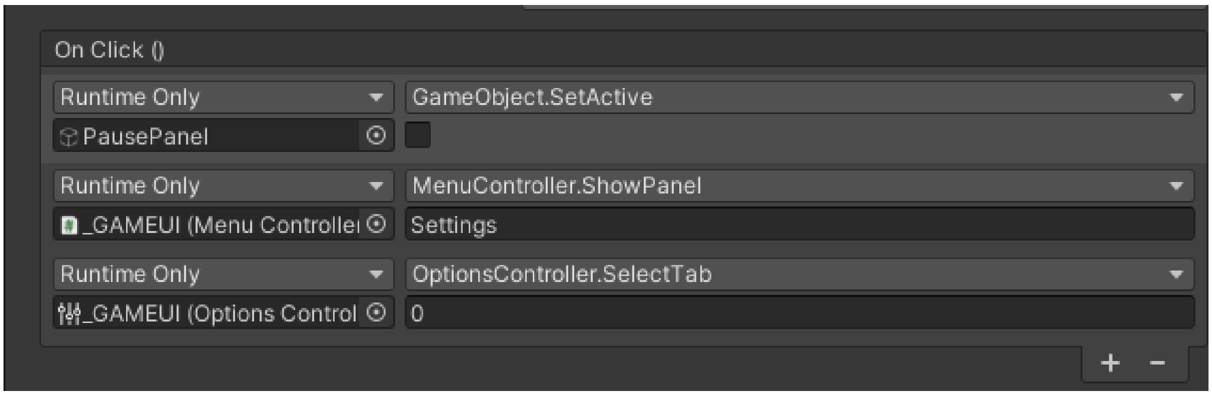
MENU CONTROLLER


Panel View
Pause Panel: PausePanel
Panel View Datas: 3
Settings
Panel Tag: Settings
Panel Object: OptionsPanel
Events: OptionsPanel (Tab Panel Events)
Gamepad Buttons: 3
Is Settings: ✓
LoadGame
Dead

- Using the Menu Controller script, you can easily define new menu panels that can be controlled using the keyboard or gamepad.


Tab Panel Events (Script)
Script: TabPanelEvents
On Cancel ()
List is Empty
On Apply ()
List is Empty
Can Invoke: ✓

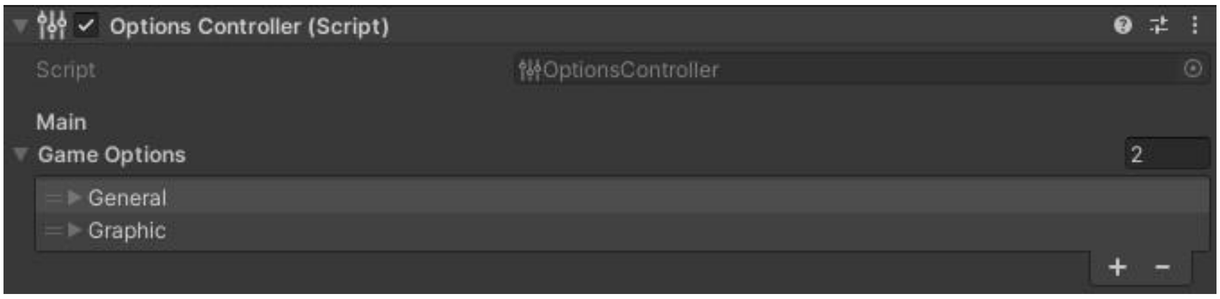
- **TabPanelEvents** defines what happens when you press **Cancel** or **Apply** button defined in the Input Action Asset. Tab Panel Events is used only when the panel is active.

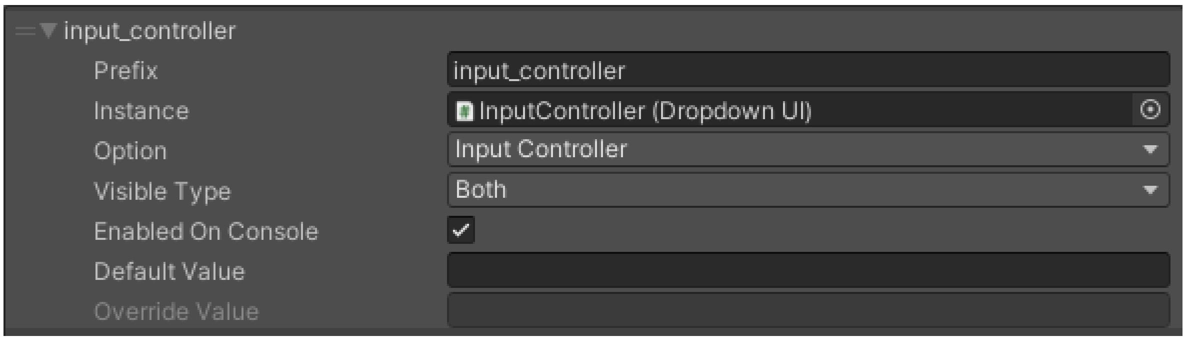

On Click ()
Runtime Only | GameObject.SetActive
PausePanel | ☐
Runtime Only | MenuController.ShowPanel
_GAMEUI (Menu Controlle | Settings
Runtime Only | OptionsController.SelectTab
_GAMEUI (Options Control | 0

- To display the panel, call the **ShowPanel()** function.
- When returning to the general menu, it is necessary to call the **ShowGeneralMenu()** and **ResetPanels()** functions.

OPTIONS CONTROLLER

- The Options Controller script gives you methods to define new game settings, which may vary depending on the current platform.


Options Controller (Script)
Script: OptionsController
Main
Game Options: 2
General
Graphic


input_controller
Prefix: input_controller
Instance: InputController (Dropdown UI)
Option: Input Controller
Visible Type: Both
Enabled On Console: ✓
Default Value:
Override Value:

- You can select one of the default option types, or select a custom option type that you can manage using the Options Controller events and functions to geta value from an option.
- You can also specify whether the option can be displayed and used on the keyboard or gamepad controller.
- To listen for an options changes, use the OnOptionsUpdated event.

```
OptionsController.OnOptionsUpdated += OnOptionsUpdated;
```

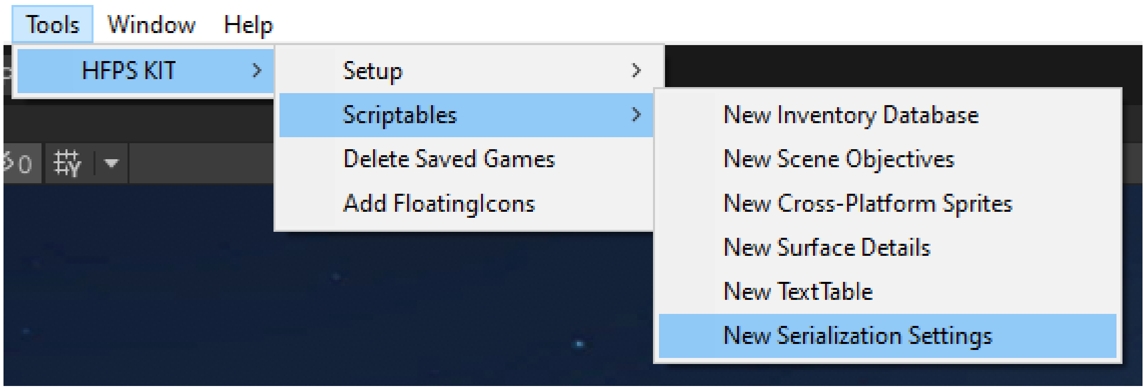
- After adding a function for the event, you can use the GetOptionValue() function to get the value of the option.

```
private void OnOptionsUpdated()
{
    if (OptionsController.GetOptionValue("KEY", out float value))
    {
        //your code goes here
    }
}
```

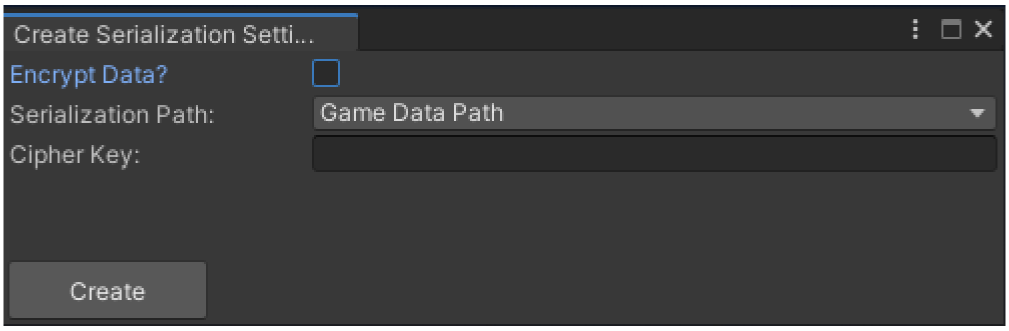
SAVE/LOAD MANAGER

SETTING UP

1. Create new **Serialization Settings** asset.



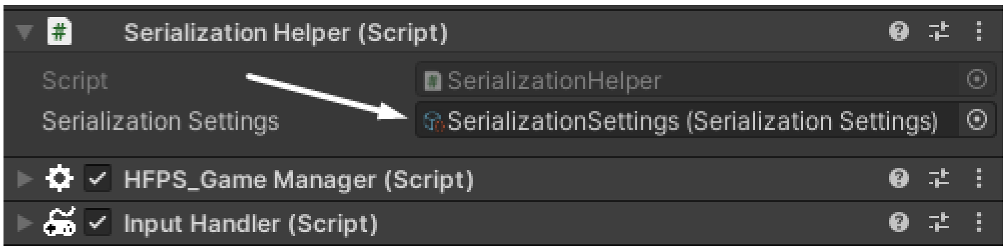
- This will show up a **Serialization Settings** window.



Create Serialization Setti...
Encrypt Data?
Serialization Path: Game Data Path
Cipher Key:

- **Cipher Key** can be a random word or text. It is used to **Save** and **Load** encrypted **Game Data**.

2. Assign your new **Serialization Settings** into **Serialization Helper** script.



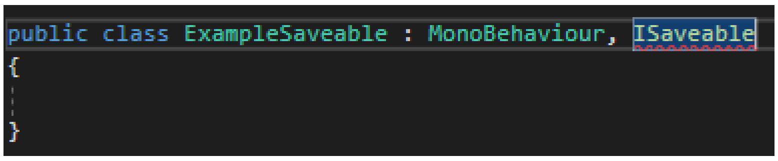
Serialization Helper (Script)
Script: SerializationHelper
Serialization Settings: SerializationSettings (Serialization Settings)
HFPS_Game Manager (Script)
Input Handler (Script)

- The serialization path is used as the default path for storing saved games or settings files.

ADDING CUSTOM SAVEABLES

There are 3 methods, how you can define new **Saveables**.

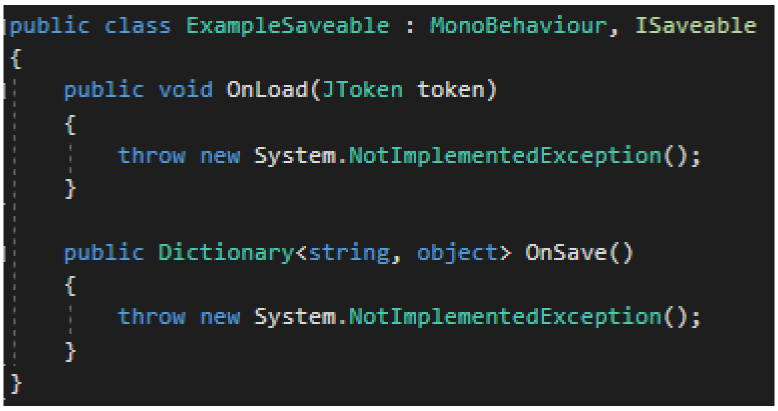
1. Adding **ISaveable** interface.



```
public class ExampleSaveable : MonoBehaviour, ISaveable
{

}
```

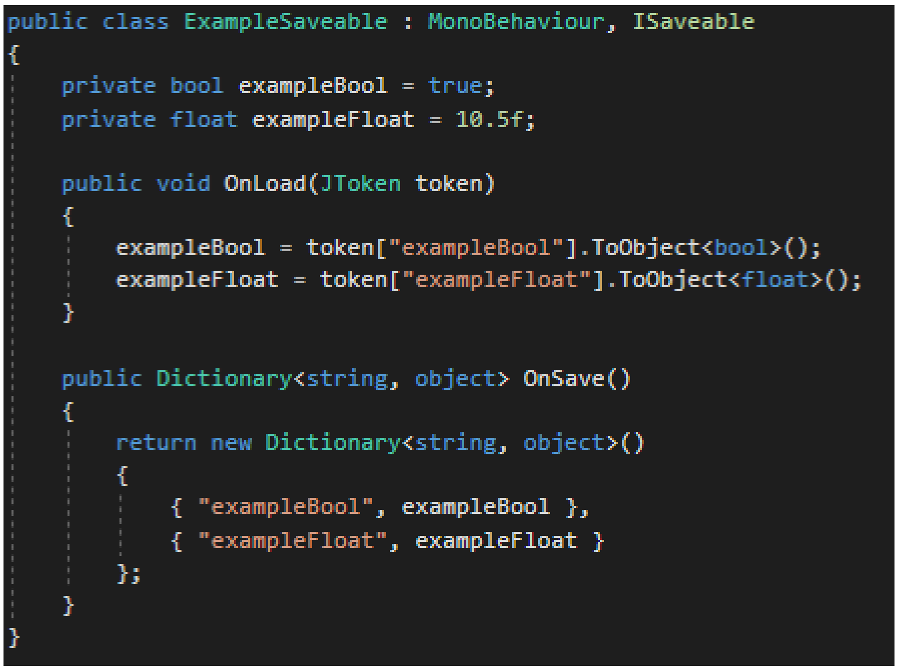
- After adding the **ISaveable** interface, you need to create the required functions.
- It can be done by pressing (**ALT + Enter or Ctrl + .**) in **Visual Studio**.



```
public class ExampleSaveable : MonoBehaviour, ISaveable
{
    public void OnLoad(JToken token)
    {
        throw new System.NotImplementedException();
    }

    public Dictionary<string, object> OnSave()
    {
        throw new System.NotImplementedException();
    }
}
```

- This will create two new functions, **OnLoad** and **OnSave**.
- These functions are the main functions for **Saving** and **Loading** game data.



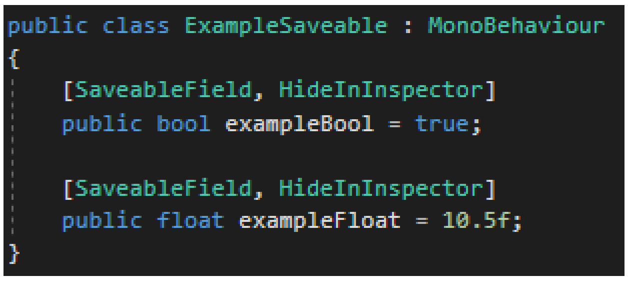
```
public class ExampleSaveable : MonoBehaviour, ISaveable
{
    private bool exampleBool = true;
    private float exampleFloat = 10.5f;

    public void OnLoad(JToken token)
    {
        exampleBool = token["exampleBool"].ToObject<bool>();
        exampleFloat = token["exampleFloat"].ToObject<float>();
    }

    public Dictionary<string, object> OnSave()
    {
        return new Dictionary<string, object>()
        {
            { "exampleBool", exampleBool },
            { "exampleFloat", exampleFloat }
        };
    }
}
```

- The **Token Path** must be the same as the **Key**, you use to store the data.
- **Dictionary Values** can be **Nested**.

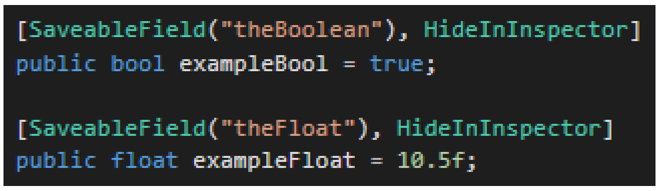
2. Adding **SaveableField** attribute to the field that you want to save.



```
public class ExampleSaveable : MonoBehaviour
{
    [SaveableField, HideInInspector]
    public bool exampleBool = true;

    [SaveableField, HideInInspector]
    public float exampleFloat = 10.5f;
}
```

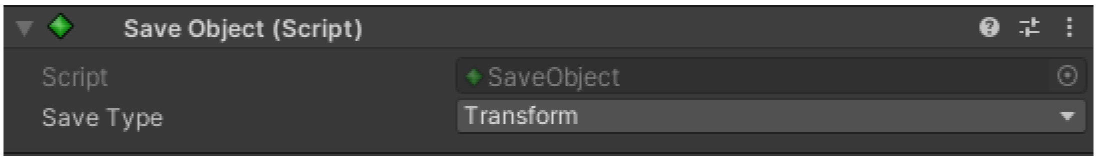
- You can also set your own field key.



```
[SaveableField("theBoolean"), HideInInspector]
public bool exampleBool = true;

[SaveableField("theFloat"), HideInInspector]
public float exampleFloat = 10.5f;
```

3. Adding **SaveObject.cs** script to **GameObject**.



Save Object (Script)

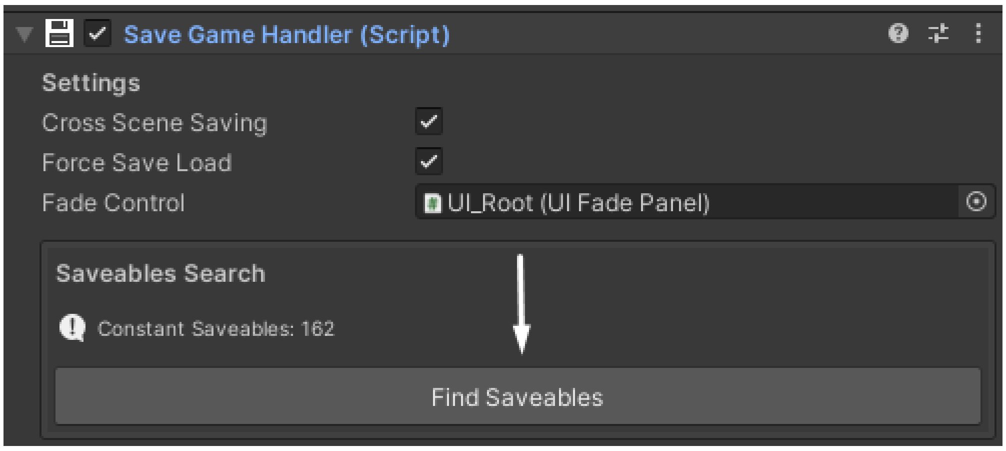
Script: SaveObject

Save Type: Transform

- This script allows you to define what you want to save from the object.
- **Supported Types:** Transform, Rigidbody, Position, Rotation and Object/Renderer Active.

SAVING GAME DATA

- By clicking the **Find Saveables** button, the script will find all **Saveables** definedby the methods in the **Adding Custom Saveables** section.



Save Game Handler (Script)

Settings

Cross Scene Saving ✓

Force Save Load ✓

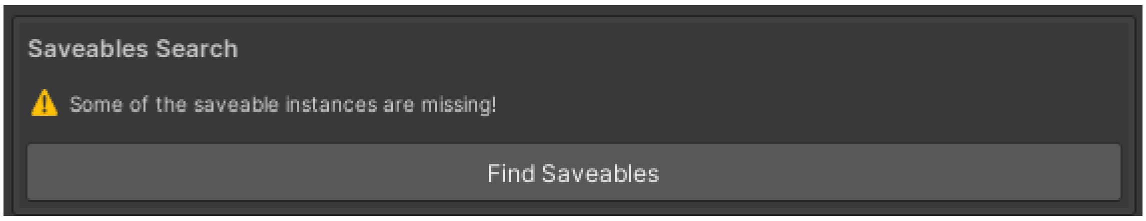
Fade Control: UI_Root (UI Fade Panel)

Saveables Search

Constant Saveables: 162

- To load a new scene and keep all inventory and equipped items, you must use the **LoadNextScene()** function located in the **HFPS_GameManager** script.
- There is a script called **NextLevelTrigger** that you can use to load next scene without losing the inventory or equipment by going on a trigger.

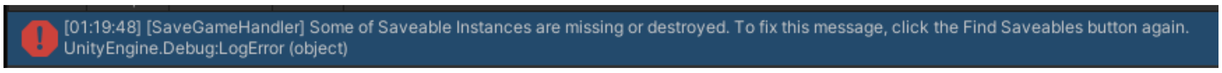
SAVEGAMEHANDLER ERROR MESSAGES



Saveables Search

Some of the saveable instances are missing!

- The referenced **Saveable** has been removed from the scene.
- You must click **Find Savablebles** again to clear this message.

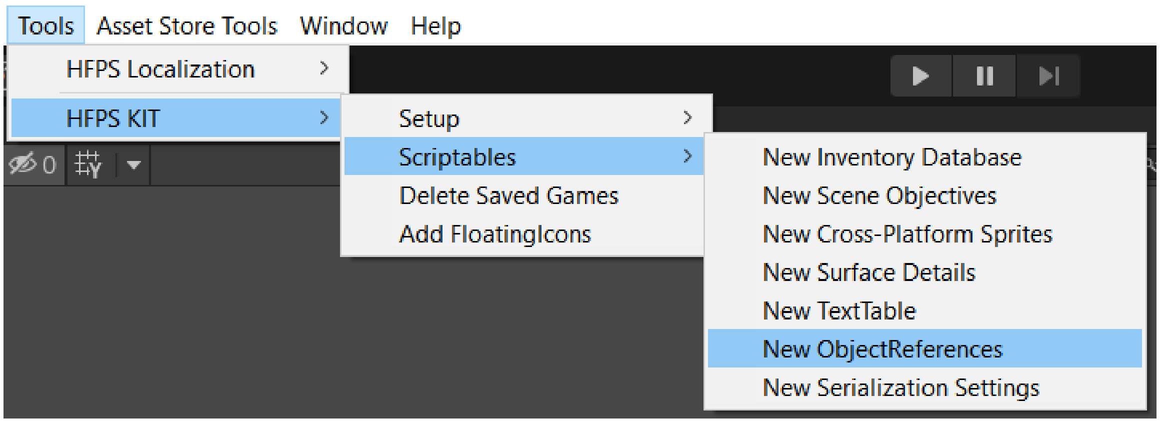


```
[01:19:48] [SaveGameHandler] Some of Saveable Instances are missing or destroyed. To fix this message, click the Find Saveables button again.
UnityEngine.Debug:LogError (object)
```

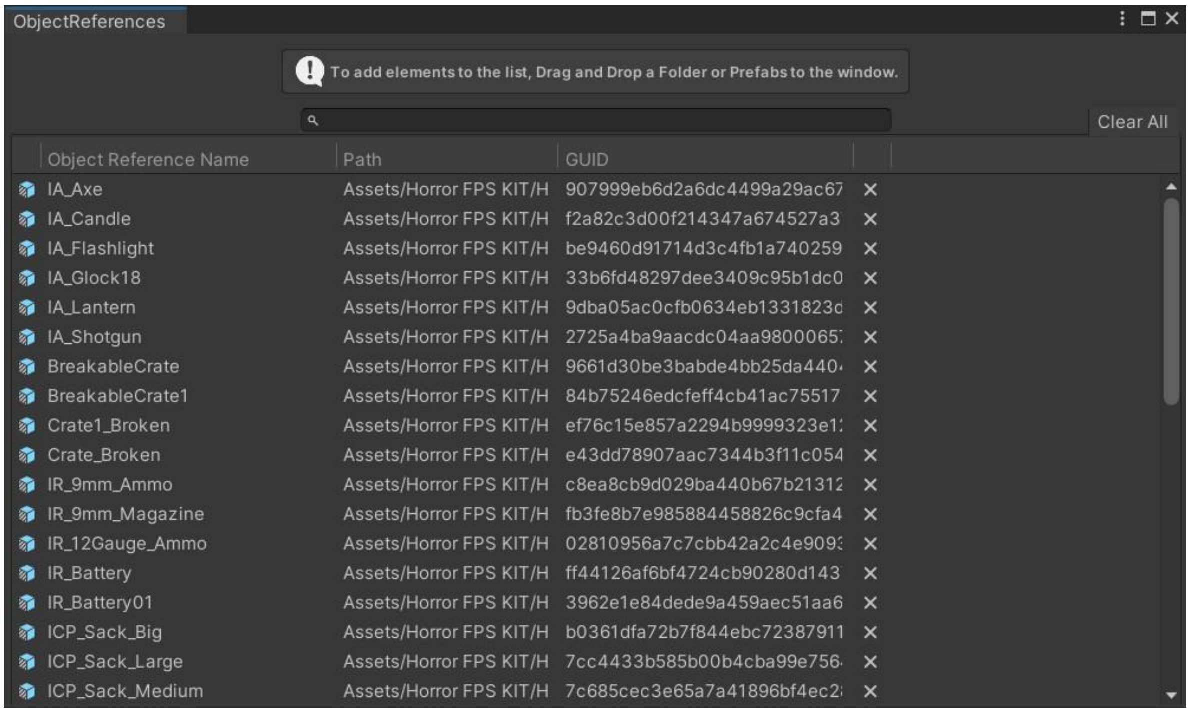
- The script loaded a **Saveable** that was saved in a file but was removed from the scene.
- You must click **Find Savablebles** again to clear this message.
- If the **Force Save Load** option is turned off, loading will be prevented if this error message is shown.

ADDING RUNTIME SAVEABLES

1. Create or open an existing **Object References** asset.



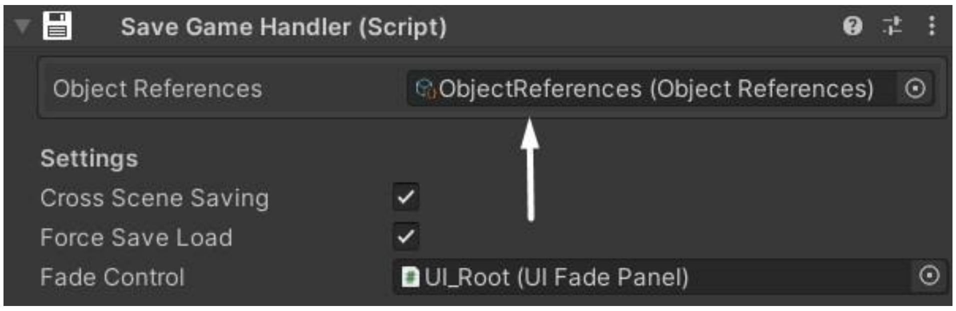
2. Open **Object References Window** and add instantiable prefabs by dragging folders or prefabs to the window.



To add elements to the list, Drag and Drop a Folder or Prefabs to the window.

Object Reference Name	Path	GUID	
IA_Axe	Assets/Horror FPS KIT/H	907999eb6d2a6dc4499a29ac67	✕
IA_Candle	Assets/Horror FPS KIT/H	f2a82c3d00f214347a674527a3	✕
IA_Flashlight	Assets/Horror FPS KIT/H	be9460d91714d3c4fb1a740259	✕
IA_Glock18	Assets/Horror FPS KIT/H	33b6fd48297dee3409c95b1dc0	✕
IA_Lantern	Assets/Horror FPS KIT/H	9dba05ac0cfb0634eb1331823d	✕
IA_Shotgun	Assets/Horror FPS KIT/H	2725a4ba9aacdc04aa9800065	✕
BreakableCrate	Assets/Horror FPS KIT/H	9661d30be3babde4bb25da440	✕
BreakableCrate1	Assets/Horror FPS KIT/H	84b75246edcfeff4cb41ac75517	✕
Crate1_Broken	Assets/Horror FPS KIT/H	ef76c15e857a2294b9999323e1	✕
Crate_Broken	Assets/Horror FPS KIT/H	e43dd78907aac7344b3f11c054	✕
IR_9mm_Ammo	Assets/Horror FPS KIT/H	c8ea8cb9d029ba440b67b21312	✕
IR_9mm_Magazine	Assets/Horror FPS KIT/H	fb3fe8b7e985884458826c9cfa4	✕
IR_12Gauge_Ammo	Assets/Horror FPS KIT/H	02810956a7c7cbb42a2c4e909	✕
IR_Battery	Assets/Horror FPS KIT/H	ff44126af6bf4724cb90280d143	✕
IR_Battery01	Assets/Horror FPS KIT/H	3962e1e84dede9a459aec51aa6	✕
ICP_Sack_Big	Assets/Horror FPS KIT/H	b0361dfa72b7f844ebc72387911	✕
ICP_Sack_Large	Assets/Horror FPS KIT/H	7cc4433b585b00b4cba99e756	✕
ICP_Sack_Medium	Assets/Horror FPS KIT/H	7c685cec3e65a7a41896bf4ec2	✕

3. Assign **Object References** asset to **SaveGameHandler** script. This will allow you to open the object reference selection tool.



Object References: ObjectReferences (Object References)

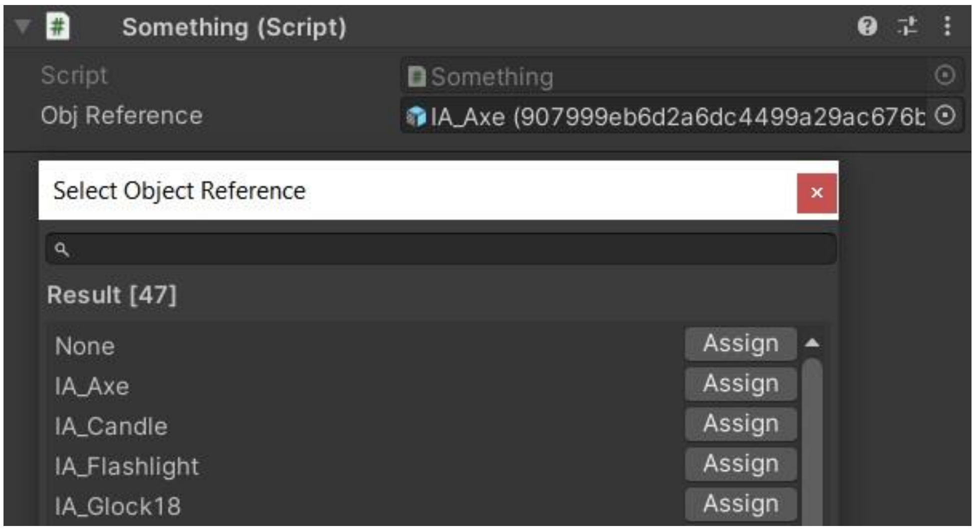
Settings
Cross Scene Saving ✓
Force Save Load ✓
Fade Control: UI_Root (UI Fade Panel)

4. Add the **ObjectReference** variable.

```
using HFPS.System;

public class Something : MonoBehaviour
{
    public ObjectReference ObjReference;
}
```

5. Assign **Object Reference** object by opening the **Object Reference Selection Tool**.


Something (Script)

Script: Something

Obj Reference: IA_Axe (907999eb6d2a6dc4499a29ac676b

Select Object Reference

Result [47]

None	Assign
IA_Axe	Assign
IA_Candle	Assign
IA_Flashlight	Assign
IA_Glock18	Assign

- This variable allows you to select the **Object Reference** defined in the first steps.

6. After selecting Object Reference, you can instantiate it at runtime using the **InstantiateSaveable()** function located in the **SaveGameHandler** script.

```
void Start()
{
    SaveGameHandler handler = SaveGameHandler.Instance;
    GameObject go = handler.InstantiateSaveable(ObjectReference, position, rotation);

    //go represents the instantiated object
}
```

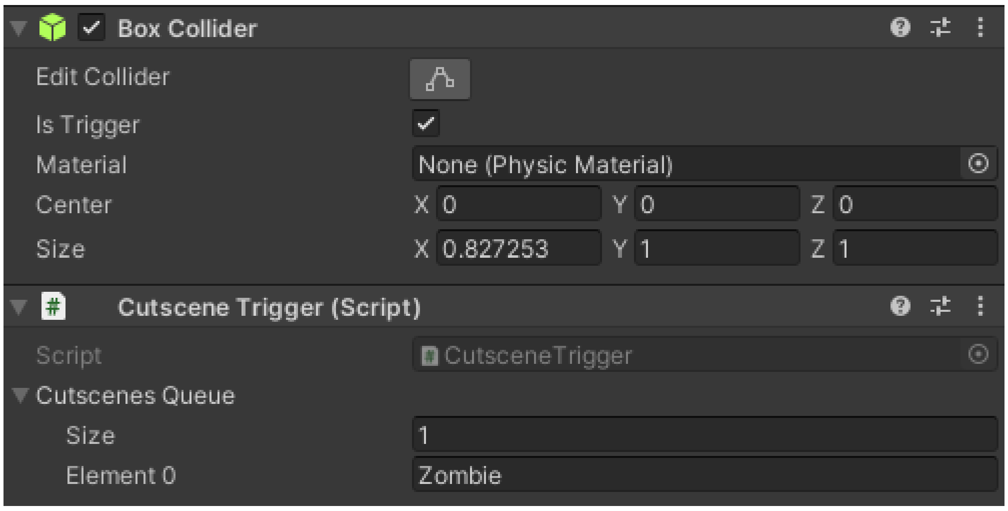
- Using the Object References we can store a specific GUID of the asset which we can use to retrieve an object that was created at runtime.

ADDING NEW CUTSCENES

1. Create a new **Timeline Animation**. You can follow the tutorial from **Brackeys**.

https://www.youtube.com/watch?v=G_uBFM3YUF4

2. Create a **Cutscene Trigger**.



Box Collider

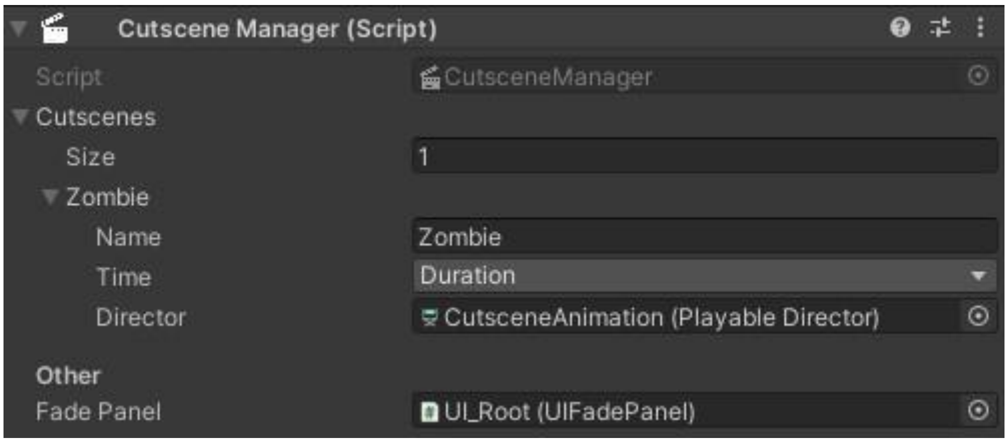
Edit Collider	
Is Trigger	✓
Material	None (Physic Material)
Center	X 0 Y 0 Z 0
Size	X 0.827253 Y 1 Z 1

Cutscene Trigger (Script)

Script	CutsceneTrigger
Cutscenes Queue	
Size	1
Element 0	Zombie

- You can **queue** the cutscenes by adding more **Elements**.

3. Assign your new **Cutscenes** to the **Cutscene Manager** script.



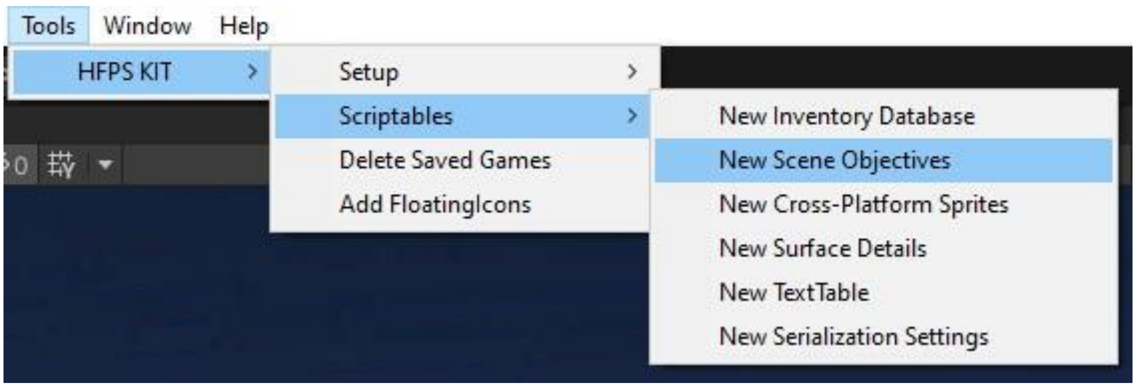
Cutscene Manager (Script)

Script	CutsceneManager
Cutscenes	
Size	1
Zombie	
Name	Zombie
Time	Duration
Director	CutsceneAnimation (Playable Director)
Other	
Fade Panel	UI_Root (UIFadePanel)

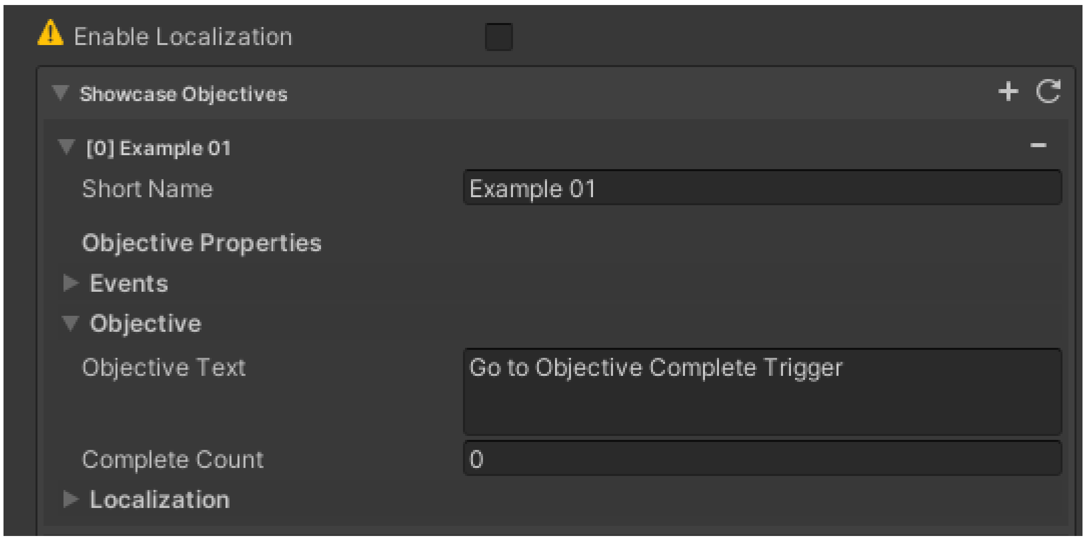
- If you set the **Time** parameter to **Manual**, you must manually skip or cancel the cutscene sequence by invoking the **Cutscene Manager -> SkipCutscene()** or **AbortCutscenes()**.

ADDING NEW OBJECTIVES

1. Create or open an existing **Scene Objectives**



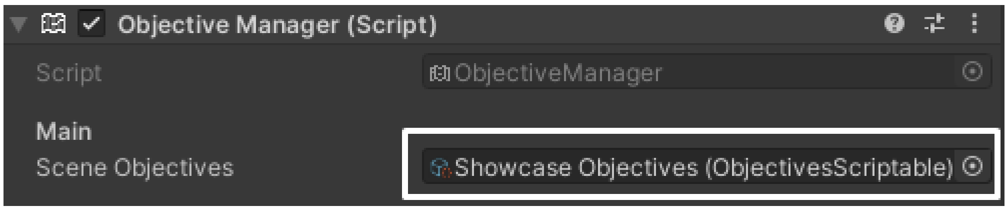
2. Clicking the **+** button you will add new objective to the **Scene Objectives** asset.



Short Name: Example 01
Objective Text: Go to Objective Complete Trigger
Complete Count: 0

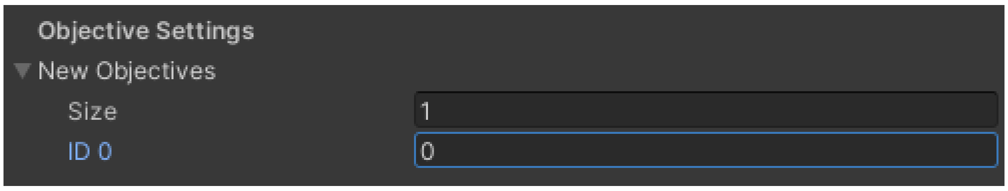
- If you reorder the objectives, clicking the refresh button will refresh the objectives IDs.
- Each objective has a unique ID that allows you to easily define which objective you want to run. Unique is a number placed before the name of the objectives: **[0]** Example 01.
- As you may have noticed, there are settings that allows localization. By clicking on the **Enable Localization** button, the system will allow you to selectthe localization key, however the **HFPS Localization System** package is required, otherwise the localization will not work!

3. Remember to always assign objectives of your scene in the **Objectives Manager** script.

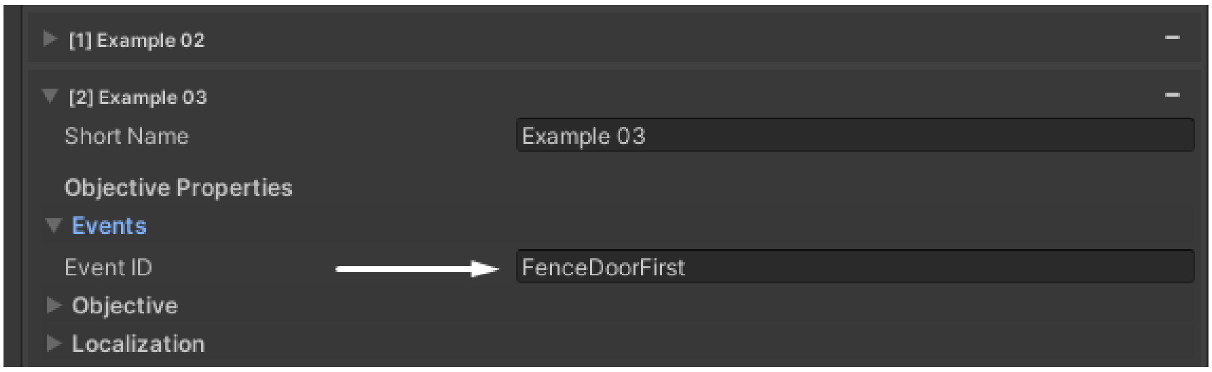


Script: ObjectiveManager
Scene Objectives: Showcase Objectives (ObjectivesScriptable)

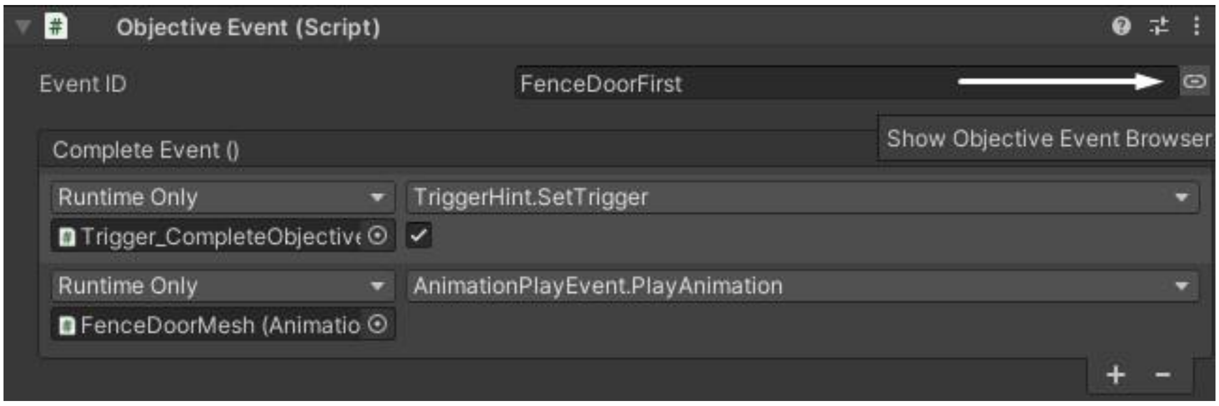
4. Create a new trigger, add **TriggerObjective.cs** script and set ID to objective which you want to run.


Objective Settings
New Objectives
Size 1
ID 0 0

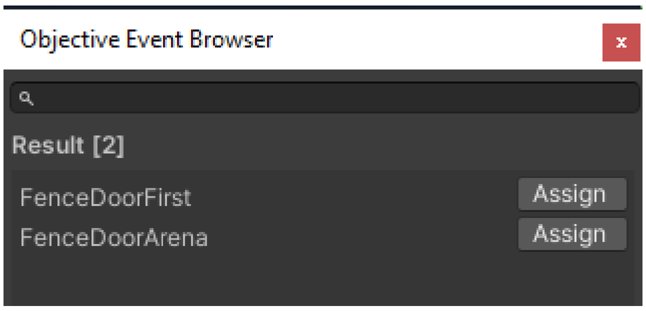
5. The Scene Objectives asset also allows you to define an event that will be called when the objective is completed.


[1] Example 02
[2] Example 03
Short Name Example 03
Objective Properties
Events
Event ID FenceDoorFirst
Objective
Localization

6. After defining an event, you must add the Objective Event script to an empty GameObject and select the event you want to use.


Objective Event (Script)
Event ID FenceDoorFirst
Show Objective Event Browser
Complete Event ()
Runtime Only | TriggerHint.SetTrigger
Trigger_CompleteObjective | ✓
Runtime Only | AnimationPlayEvent.PlayAnimation
FenceDoorMesh (Animatio

7. Clicking the **Link** button, you will display the **Objective Event Browser Window**.

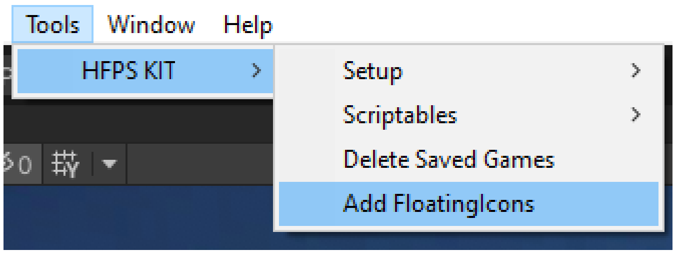

Objective Event Browser
Result [2]
FenceDoorFirst Assign
FenceDoorArena Assign

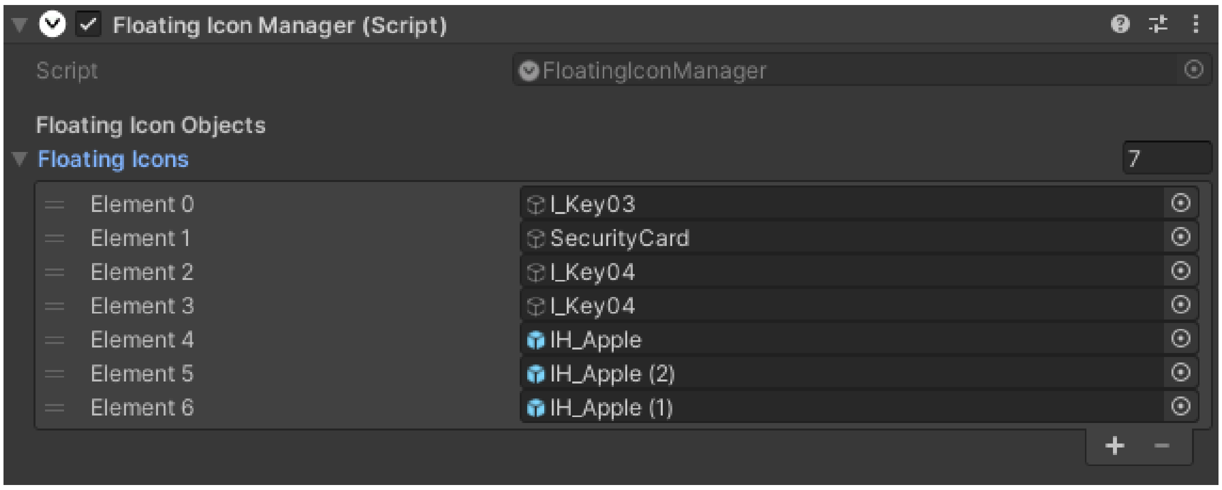
Complete Count: The objective will be fully completed if you complete the objectiven-times.

To show progress to the objective, insert "**{0}/{1}**" to the objective text.

ADDING NEW FLOATING OBJECTS

1. When you select all the **GameObjects** you want to define to display the floatingicon above object, pressing the **Add FloatingIcons** button will automatically add the objects to the list.



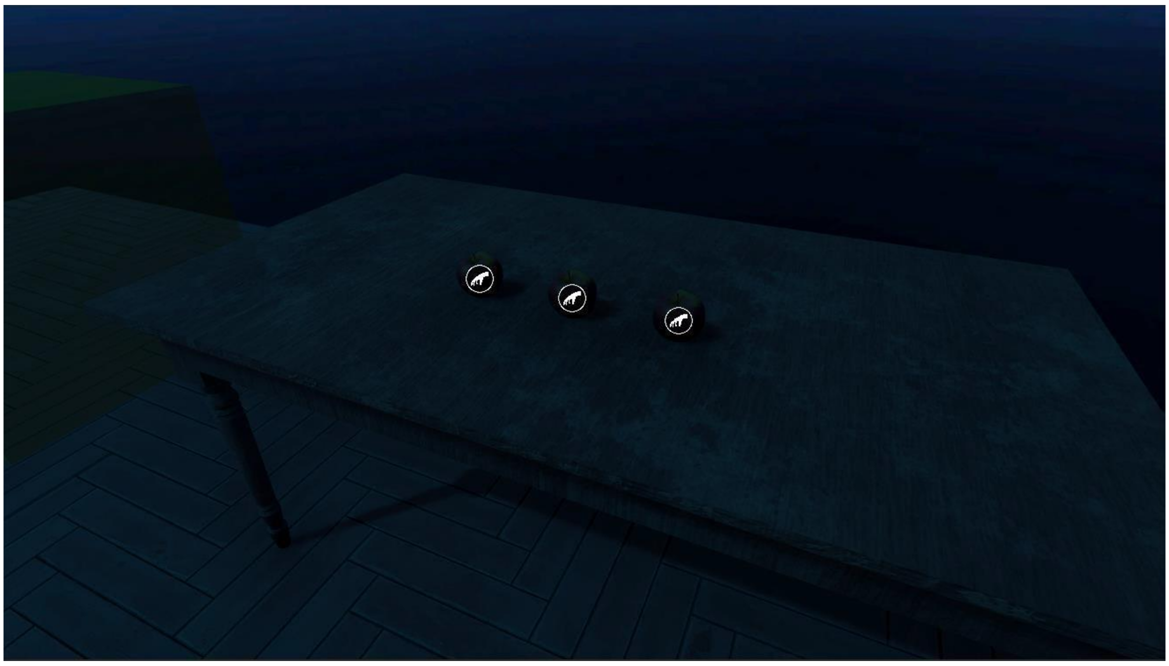

Floating Icon Manager (Script)

Script: FloatingIconManager

Floating Icon Objects

Floating Icons: 7

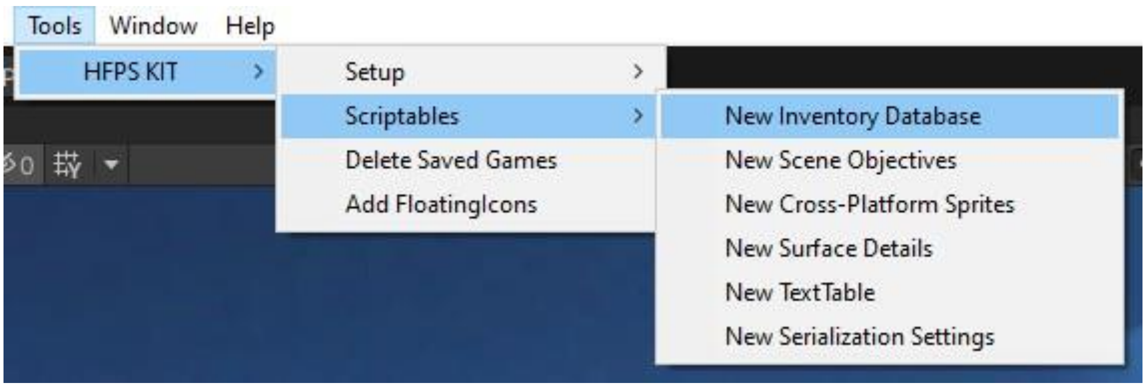
Element 0	I_Key03
Element 1	SecurityCard
Element 2	I_Key04
Element 3	I_Key04
Element 4	IH_Apple
Element 5	IH_Apple (2)
Element 6	IH_Apple (1)



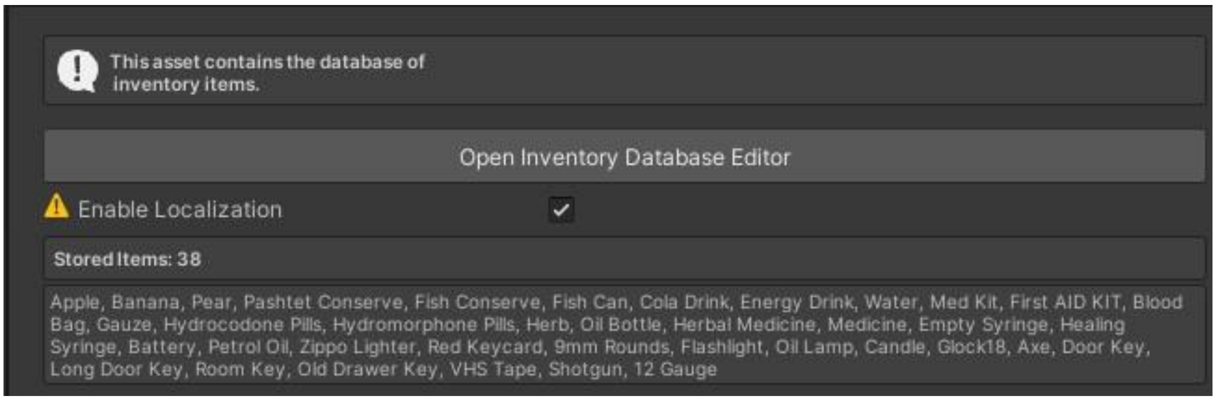
INVENTORY

ADDING NEW INVENTORY ITEMS

1. Create or Open an existing **Inventory Database** asset.



2. To open the **Inventory Database Window**, double-click the asset or click the **Open Inventory Database Editor** button.


This asset contains the database of inventory items.

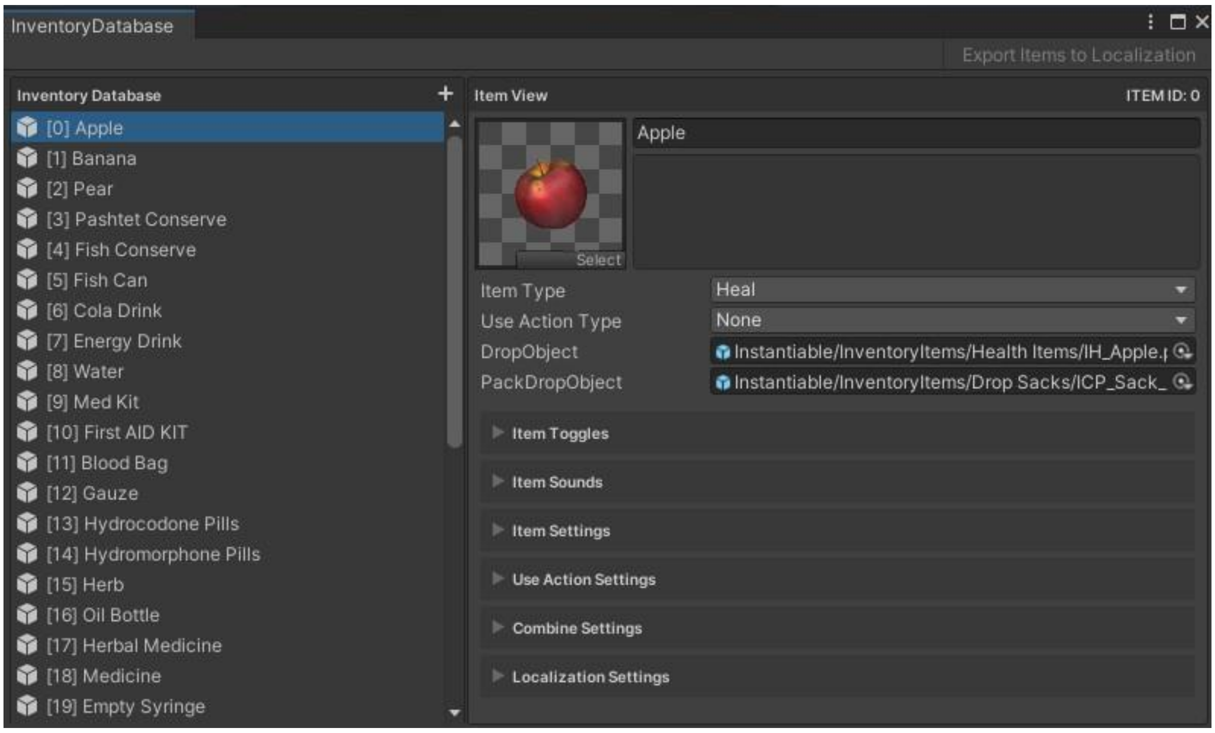
Open Inventory Database Editor

Enable Localization ✓

Stored Items: 38

Apple, Banana, Pear, Pashtet Conserve, Fish Conserve, Fish Can, Cola Drink, Energy Drink, Water, Med Kit, First AID KIT, Blood Bag, Gauze, Hydrocodone Pills, Hydromorphone Pills, Herb, Oil Bottle, Herbal Medicine, Medicine, Empty Syringe, Healing Syringe, Battery, Petrol Oil, Zippo Lighter, Red Keycard, 9mm Rounds, Flashlight, Oil Lamp, Candle, Glock18, Axe, Door Key, Long Door Key, Room Key, Old Drawer Key, VHS Tape, Shotgun, 12 Gauge

- This will show up the window where you can manage all inventory items.

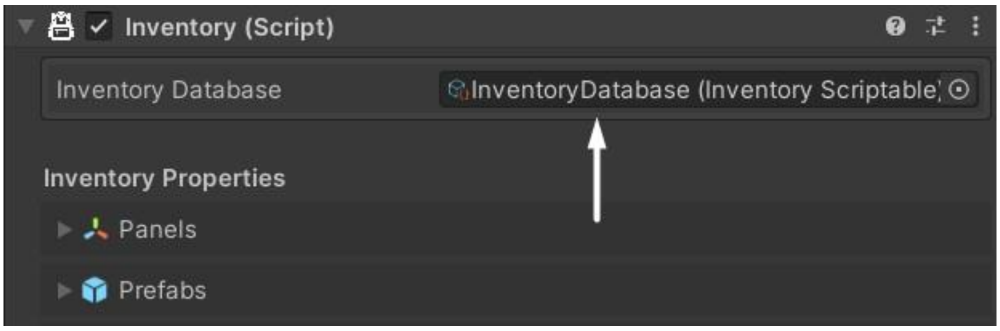


[0] Apple
[1] Banana
[2] Pear
[3] Pashtet Conserve
[4] Fish Conserve
[5] Fish Can
[6] Cola Drink
[7] Energy Drink
[8] Water
[9] Med Kit
[10] First AID KIT
[11] Blood Bag
[12] Gauze
[13] Hydrocodone Pills
[14] Hydromorphone Pills
[15] Herb
[16] Oil Bottle
[17] Herbal Medicine
[18] Medicine
[19] Empty Syringe

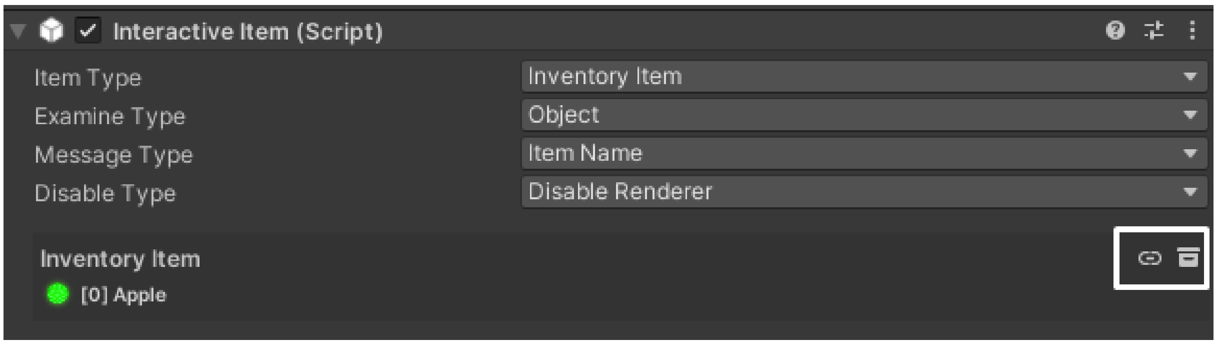
Apple

Item Type	Heal
Use Action Type	None
DropObject	Instantiable/InventoryItems/Health Items/IH_Apple.p
PackDropObject	Instantiable/InventoryItems/Drop Sacks/ICP_Sack_

3. New created **Inventory Database** must be assigned in main **Inventory** script which is at **_GAMEUI** object.

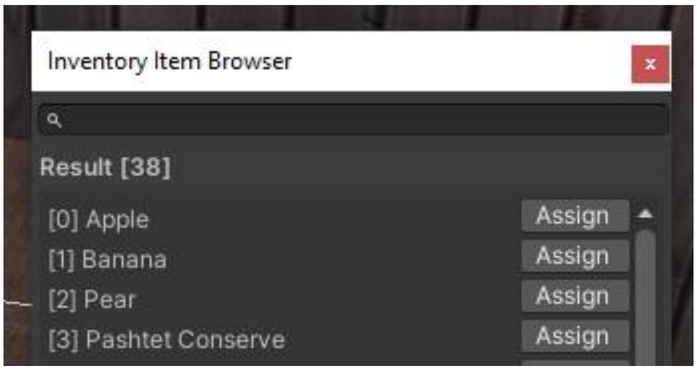

Inventory (Script)
Inventory Database: InventoryDatabase (Inventory Scriptable)
Inventory Properties
Panels
Prefabs

4. To interact with a newly created item, add the **InteractiveItem.cs** script to the object and select the item you want to use in the **Inventory Item Browser**.


Interactive Item (Script)

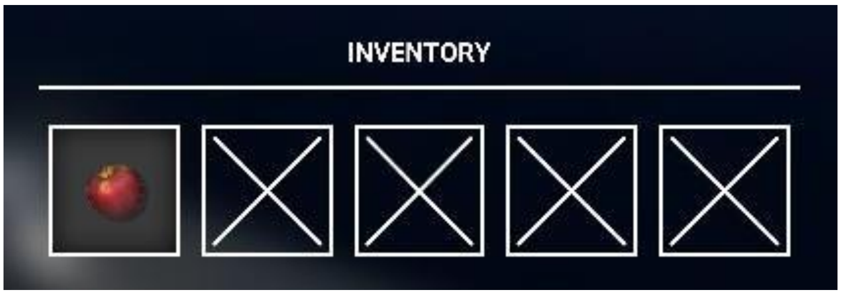
Item Type	Inventory Item
Examine Type	Object
Message Type	Item Name
Disable Type	Disable Renderer

Inventory Item
● [0] Apple


Inventory Item Browser
Result [38]

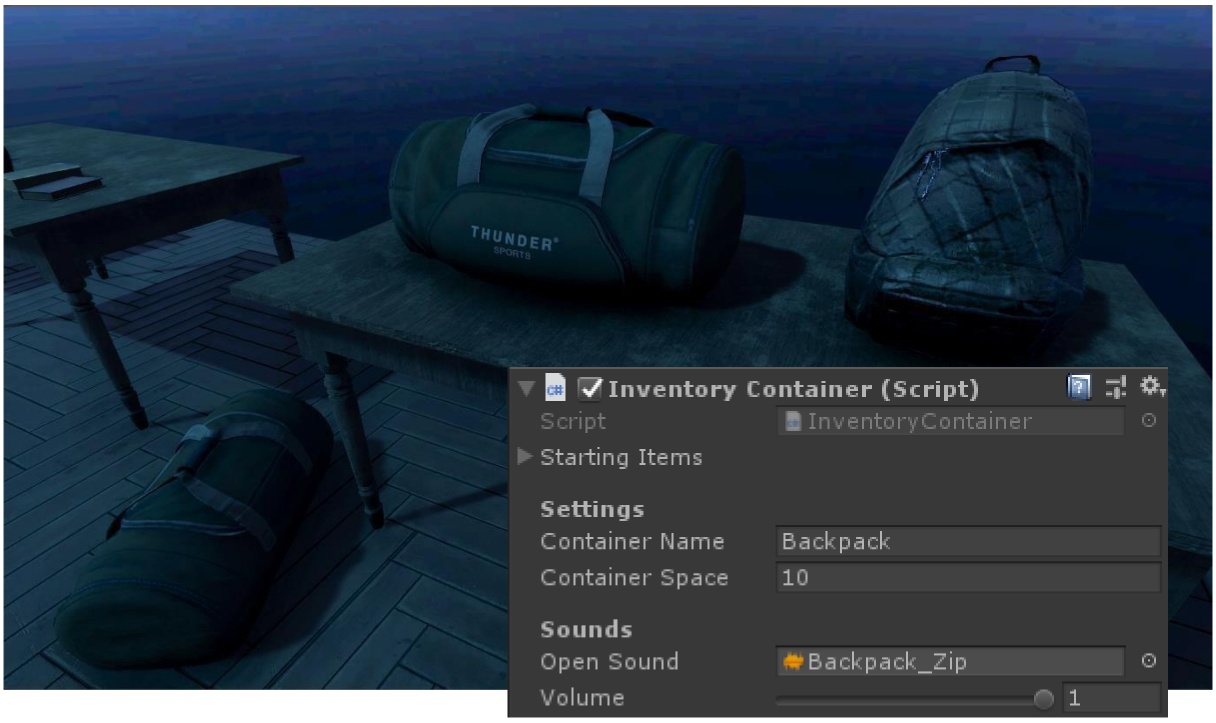
[0] Apple	Assign
[1] Banana	Assign
[2] Pear	Assign
[3] Pashtet Conserve	Assign

- The **Item ID** is read-only and is defined automatically by the database script.
- If you change the order of any item, all item **IDs** will be automatically updated with the current item order.



INVENTORY CONTAINERS

- All items from **Inventory Database** can be stored inside **Inventory Containers**.

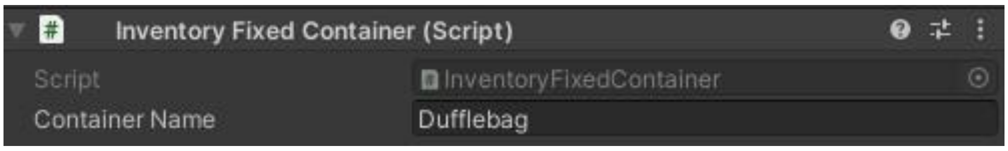


Inventory Container (Script)

Script	InventoryContainer
Starting Items	
Settings	
Container Name	Backpack
Container Space	10
Sounds	
Open Sound	Backpack_Zip
Volume	1

- To store an **Item** to a **Container**, simply add the **InventoryContainer.cs** script to it.

- You can also define **Fixed Containers** that can contain the same items as another container by adding the **InventoryFixedContainer.cs** script.

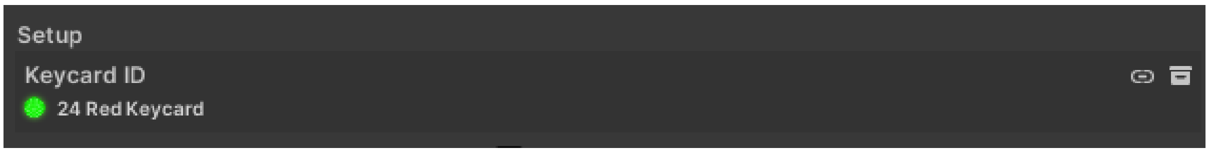


Inventory Fixed Container (Script)

Script	InventoryFixedContainer
Container Name	Dufflebag

- To convert an int field to an Inventory Selector field, you can use the **[InventorySelector]** attribute in the int field.

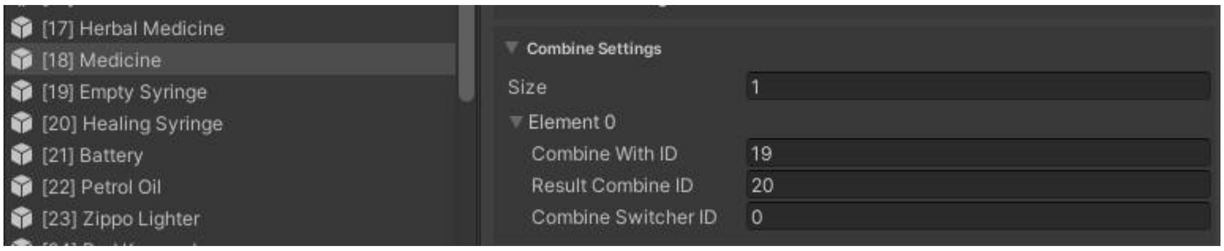
```
[InventorySelector]
public int InventoryID;
```



Setup
Keycard ID
24 Red Keycard

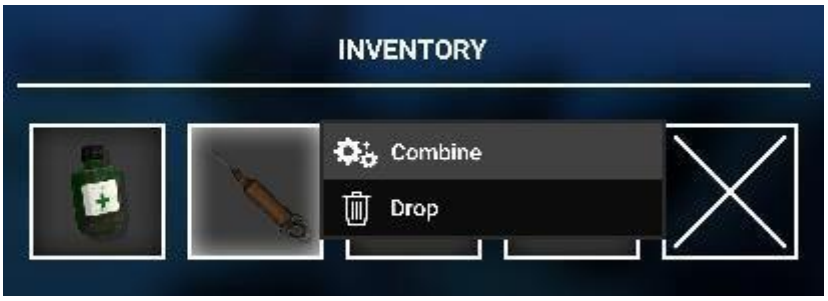
COMBINABLE ITEMS

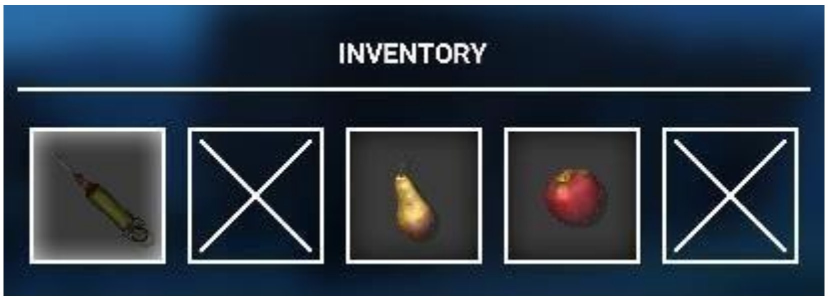
- Inventory has a feature that allows to combine two different objects to getanother useful item.
- This can be done by defining the **Combine Settings** in the Inventory Database Editor.



[17] Herbal Medicine
[18] Medicine
[19] Empty Syringe
[20] Healing Syringe
[21] Battery
[22] Petrol Oil
[23] Zippo Lighter

Combine Settings
Size 1
Element 0
Combine With ID 19
Result Combine ID 20
Combine Switcher ID 0

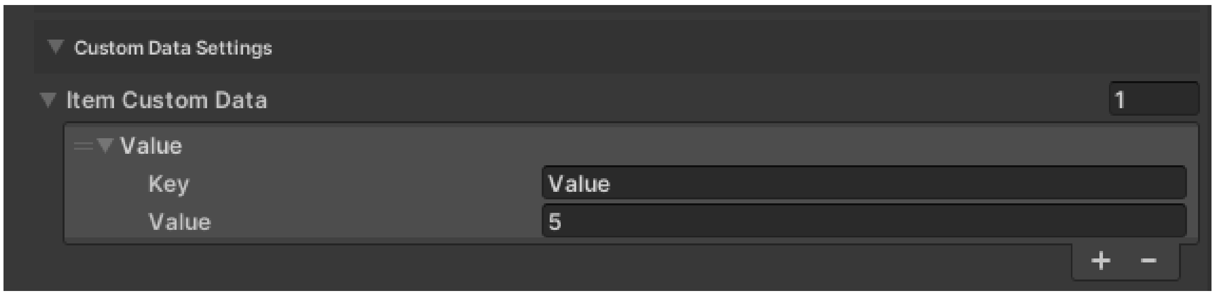




INVENTORY CUSTOM ACTIONS

- To activate **Custom Actions,** you must enable **Do Action Use** or **Do Action Combine** inside **Item Toggles** drop-down.

1. Define your **Item Custom Data** inside the **Interactive Item** script.

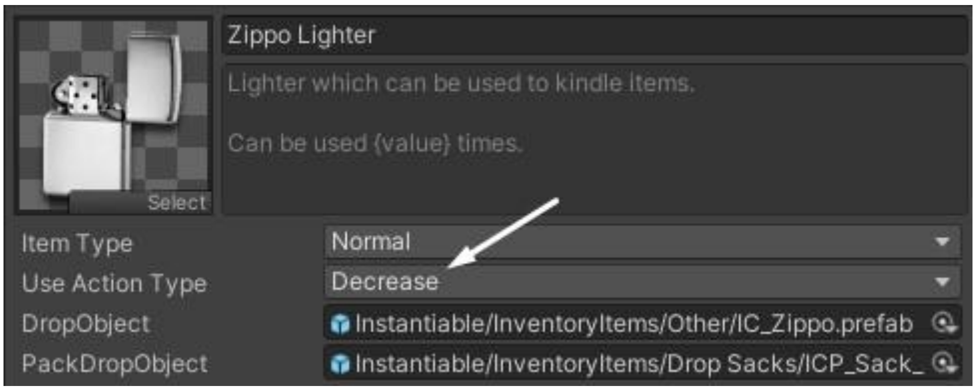


Custom Data Settings

Item Custom Data	1
Value	
Key	Value
Value	5

- **Predefined Keys**: Value, Path, Tag

2. Define **Use Action Type** in the Item Settings.



Zippo Lighter

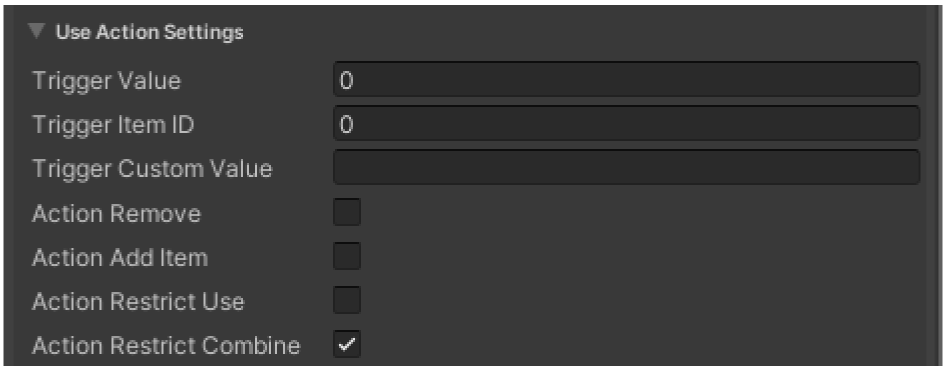
Lighter which can be used to kindle Items.

Can be used {value} times.

Item Type	Normal
Use Action Type	Decrease
DropObject	Instantiable/InventoryItems/Other/IC_Zippo.prefab
PackDropObject	Instantiable/InventoryItems/Drop Sacks/ICP_Sack_

- We set **Use Action Type** to **Decrease** to decrease the value each time zippo is combined with a candle.

3. Define **Use Action Settings**

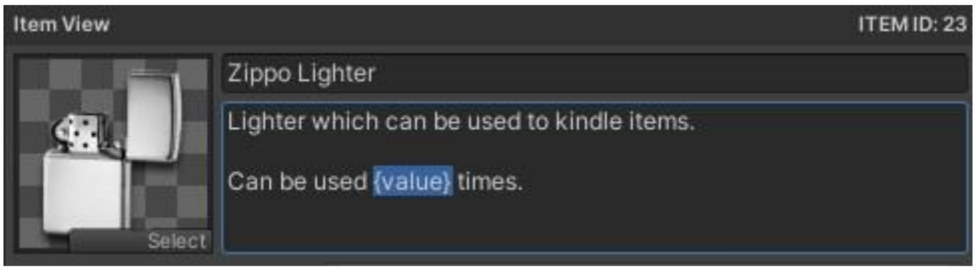


Use Action Settings

Trigger Value	0
Trigger Item ID	0
Trigger Custom Value	
Action Remove	☐
Action Add Item	☐
Action Restrict Use	☐
Action Restrict Combine	✓

- We have now defined that when the Zippo Lighter is used 5 times, you will not be able to light the candle again.

- To display the current value in the item description, you must add a {value} tag to the description text, as shown in the screenshot.


Item View ITEM ID: 23

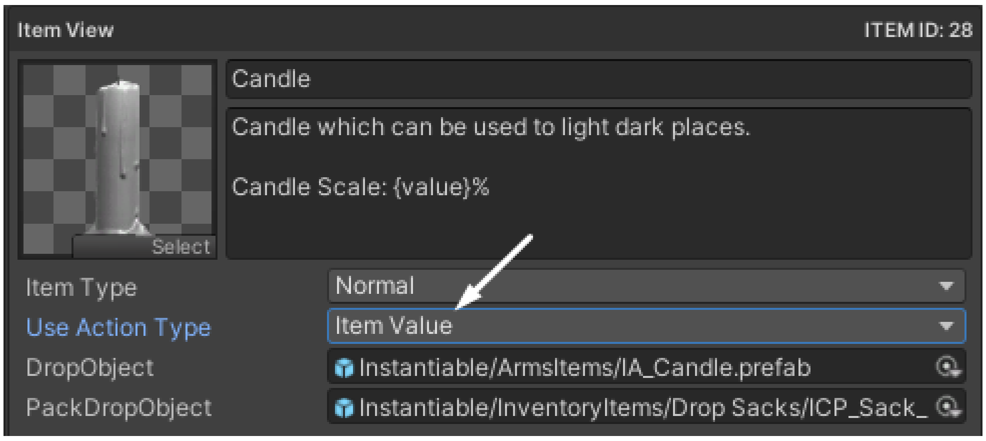
Zippo Lighter

Lighter which can be used to kindle items.

Can be used {value} times.

- If you want to use the **Item Value** action type, you can follow the same steps, but this action type is specific in that you can set the value manually by script using the **IItemValueProvider** interface.

1. Set **Use Action Type** to **Item Value**.


Item View ITEM ID: 28

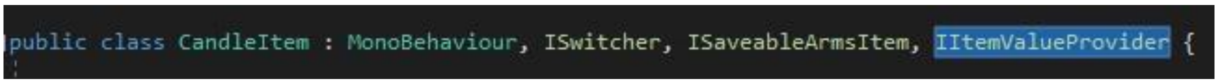
Candle

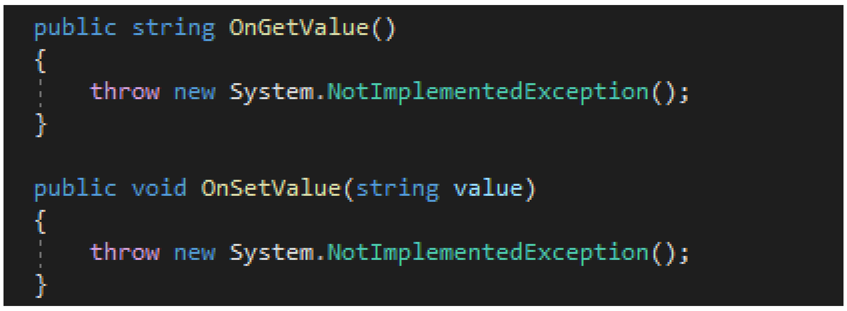
Candle which can be used to light dark places.

Candle Scale: {value}%

Item Type	Normal
Use Action Type	Item Value
DropObject	Instantiable/ArmsItems/IA_Candle.prefab
PackDropObject	Instantiable/InventoryItems/Drop Sacks/ICP_Sack_

2. Add **IItemValueProvider** interface and its functions.


```
public class CandleItem : MonoBehaviour, ISwitcher, ISaveableArmsItem, IItemValueProvider {
```

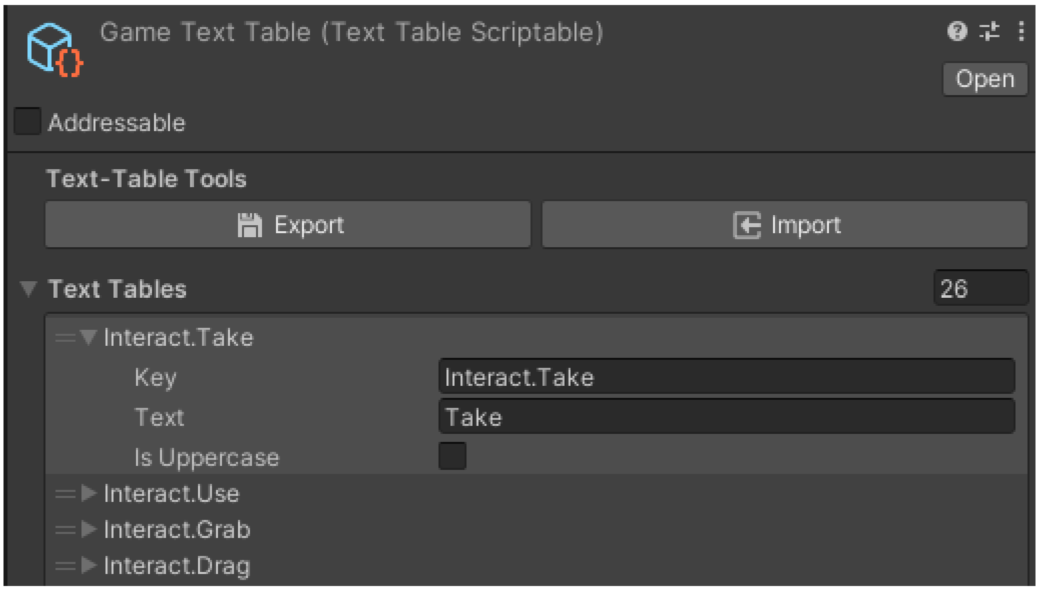

```
public string OnGetValue()
{
    throw new System.NotImplementedException();
}

public void OnSetValue(string value)
{
    throw new System.NotImplementedException();
}
```

- In the Item Toggles drop-down menu, we set the action to be applied when the item is combined. When the item is combined, OnSetValue () is called.
- **OnGetValue()** is different, you must update the value manually using **Inventory.GetItemData().data** or setting the custom data in **AddItem()** function.

CHANGING SCRIPTS UI TEXTS

- To change the UI script texts, find the **GameTextTable** asset.



Game Text Table (Text Table Scriptable)

Open

Addressable

Text-Table Tools

Export | Import

Text Tables 26

Interact.Take

Key: Interact.Take

Text: Take

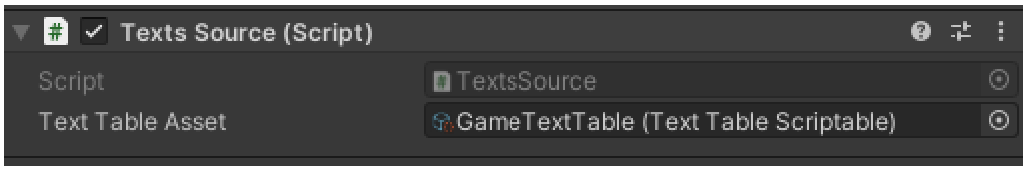
Is Uppercase

Interact.Use

Interact.Grab

Interact.Drag

- From there, you can change or create new user interface texts, which you want to use in scripts.



Texts Source (Script)

Script: TextsSource

Text Table Asset: GameTextTable (Text Table Scriptable)

- The newly created **TextTable** asset can be assigned in the **Text Source** script.
- You can easily use the TextsSource functions to retrieve text from a text table.
- The following code shows a basic example of retrieving texts.

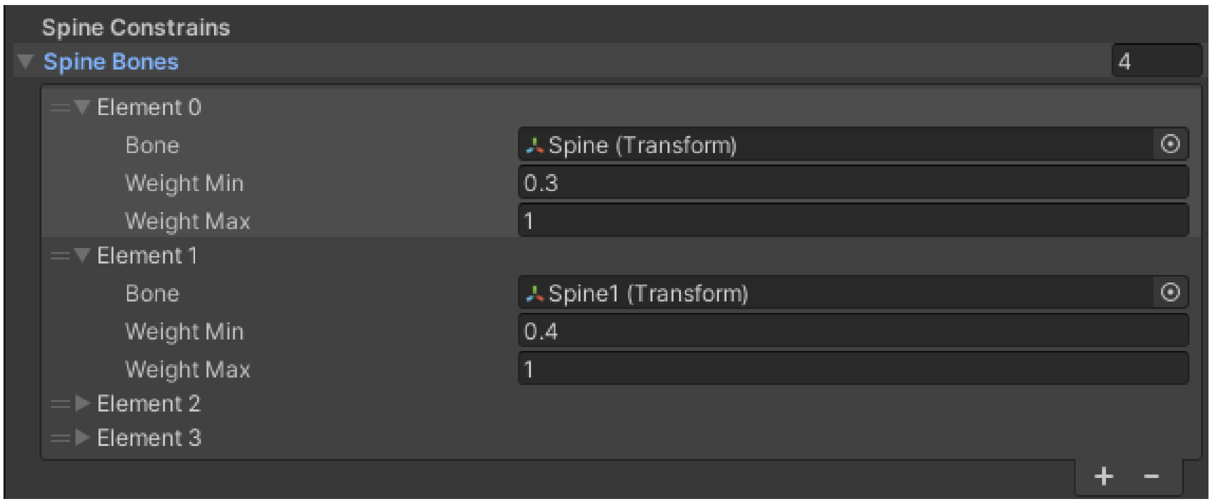
```
using HFPS.Systems;

void Awake()
{
    TextsSource.Subscribe(OnInitTexts);
    //or TextsSource.OnInitTexts += OnInitTexts;
}

private void OnInitTexts()
{
    string text = TextsSource.GetText("Interact.Take", "Take");
    Debug.Log(text);
}
```

PLAYER BODY (Body Animator)

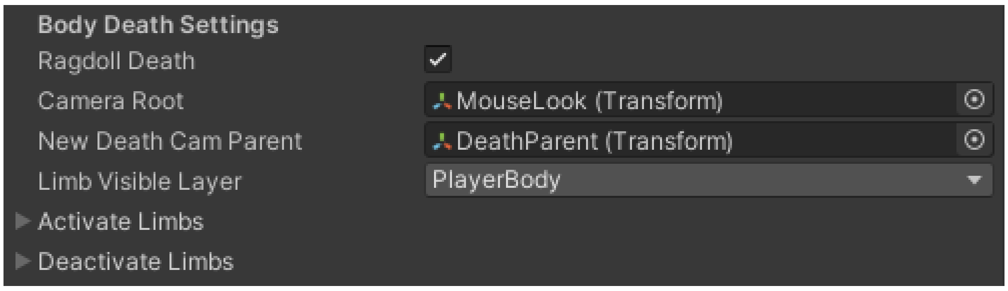
- HFPS scripts are not linked to **Body Animator**, so if you don't want to have a first-person body, you can simply delete the **HeroBody** object.



Spine Constrains

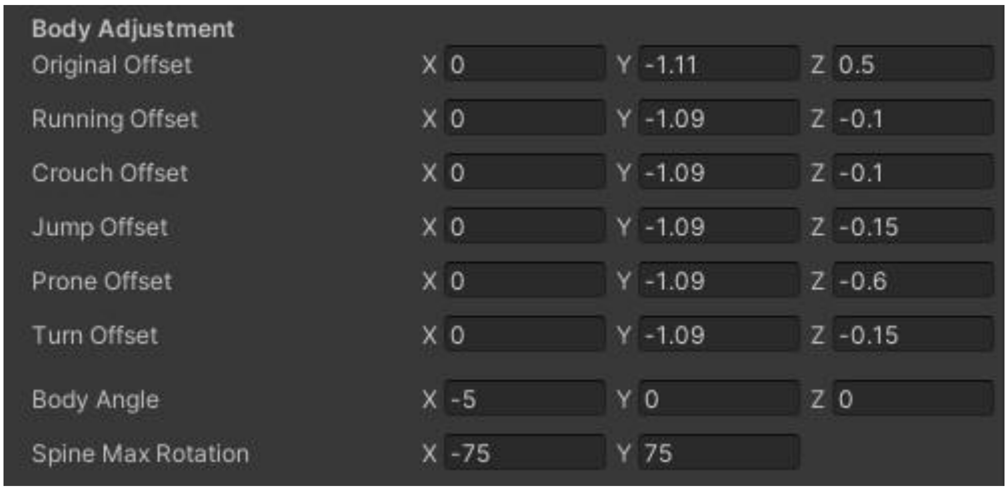
Spine Bones	4
Element 0	
Bone	Spine (Transform)
Weight Min	0.3
Weight Max	1
Element 1	
Bone	Spine1 (Transform)
Weight Min	0.4
Weight Max	1
Element 2	
Element 3	

- Using the **Spine Bones,** you can easily define the entire spine and its min/max rotation by adjusting the weights.



Body Death Settings

Ragdoll Death	✓
Camera Root	MouseLook (Transform)
New Death Cam Parent	DeathParent (Transform)
Limb Visible Layer	PlayerBody
Activate Limbs	
Deactivate Limbs	



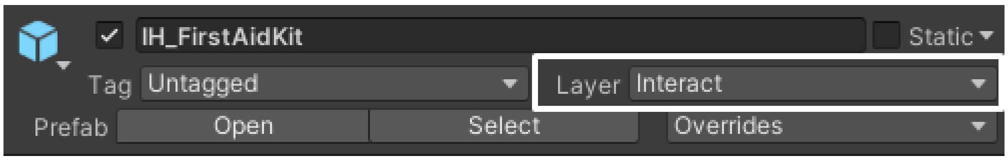
Body Adjustment

	X	Y	Z
Original Offset	0	-1.11	0.5
Running Offset	0	-1.09	-0.1
Crouch Offset	0	-1.09	-0.1
Jump Offset	0	-1.09	-0.15
Prone Offset	0	-1.09	-0.6
Turn Offset	0	-1.09	-0.15
Body Angle	-5	0	0
Spine Max Rotation	-75	75	

- You can adjust **Body Death**, **Body Adjustments** and **Other** settings dependingto your needs.

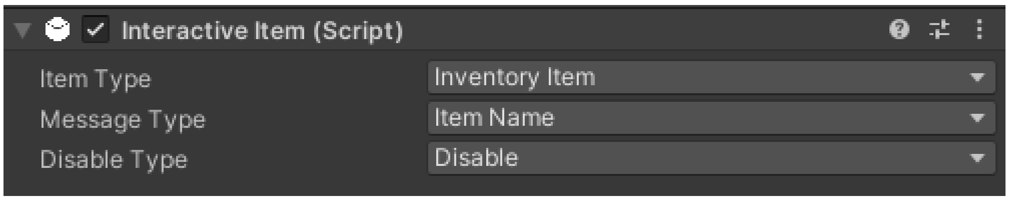
ADDING NEW INTERACTIVE ITEMS

1. Set the object layer to **Interact**.



IH_FirstAidKit — Static
Tag: Untagged — Layer: Interact
Prefab: Open | Select | Overrides

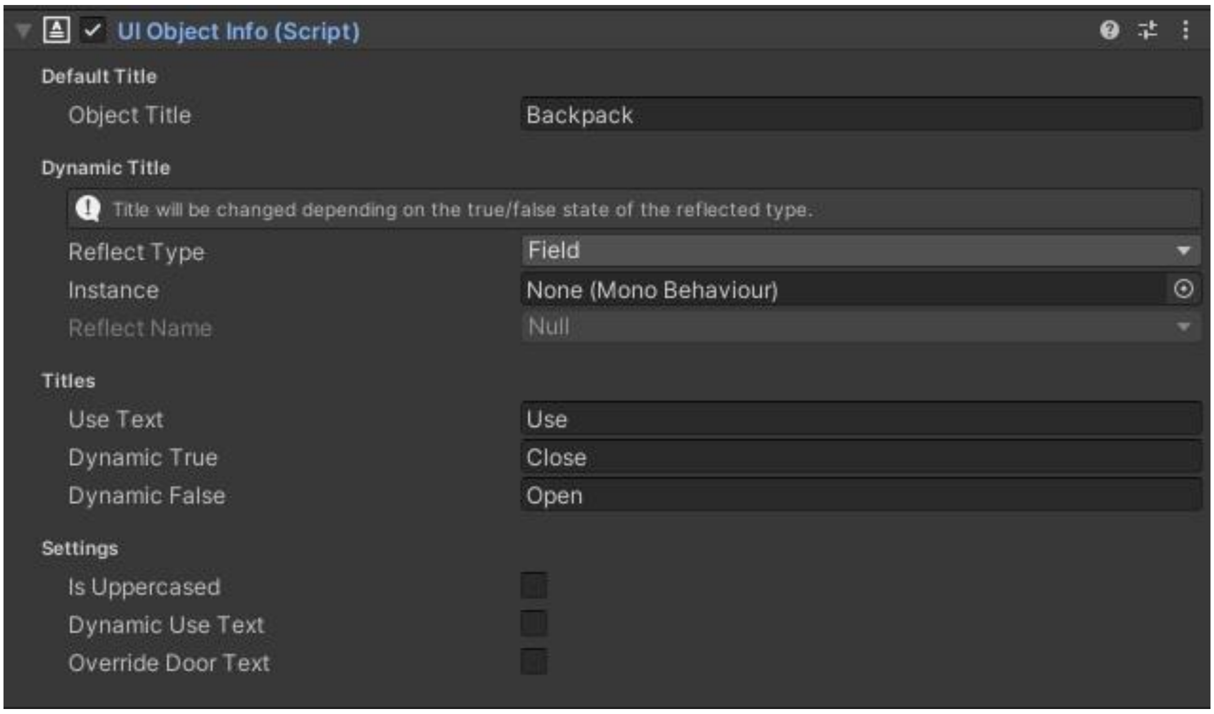
2. Add the **InteractiveItem.cs** script to the object.



Interactive Item (Script)

Item Type	Inventory Item
Message Type	Item Name
Disable Type	Disable

3. To change the user interface texts, add the **UIObjectInfo.cs** script.



UI Object Info (Script)

Default Title

Object Title	Backpack

Dynamic Title

Title will be changed depending on the true/false state of the reflected type.

Reflect Type	Field
Instance	None (Mono Behaviour)
Reflect Name	Null

Titles

Use Text	Use
Dynamic True	Close
Dynamic False	Open

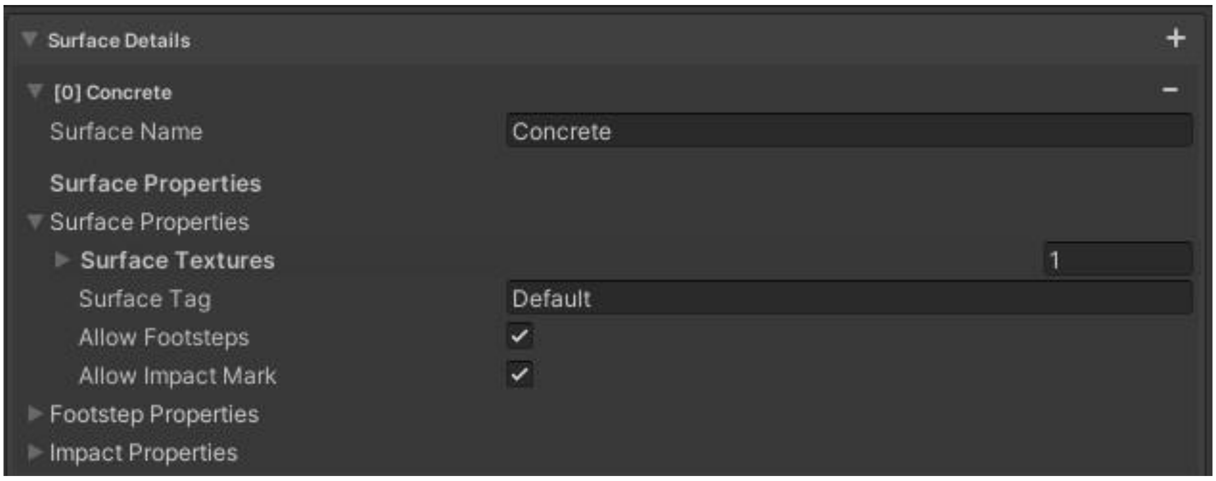
Settings

Is Uppercased	☐
Dynamic Use Text	☐
Override Door Text	☐

- To change interaction titles depending on another script field, assign **Instance** and **Reflect Name**.
- To create your own interactive object with your own script, create a public **UseObject()** function.

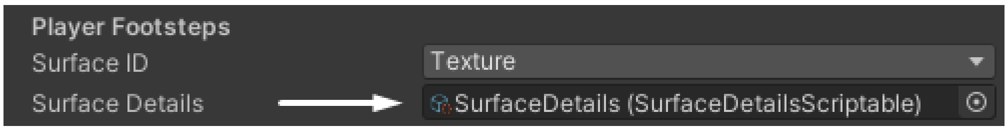
SURFACE DETAILS

- With surface details, you can easily define new materials with different material surface settings.



Surface Details
[0] Concrete
Surface Name: Concrete
Surface Properties
Surface Properties
Surface Textures: 1
Surface Tag: Default
Allow Footsteps: ✓
Allow Impact Mark: ✓
Footstep Properties
Impact Properties

- You can use **Surface Textures** to define different footsteps or bullet marks for different textures.
- **Surface Details** also support terrain textures, so you can easily define each surface of the game.

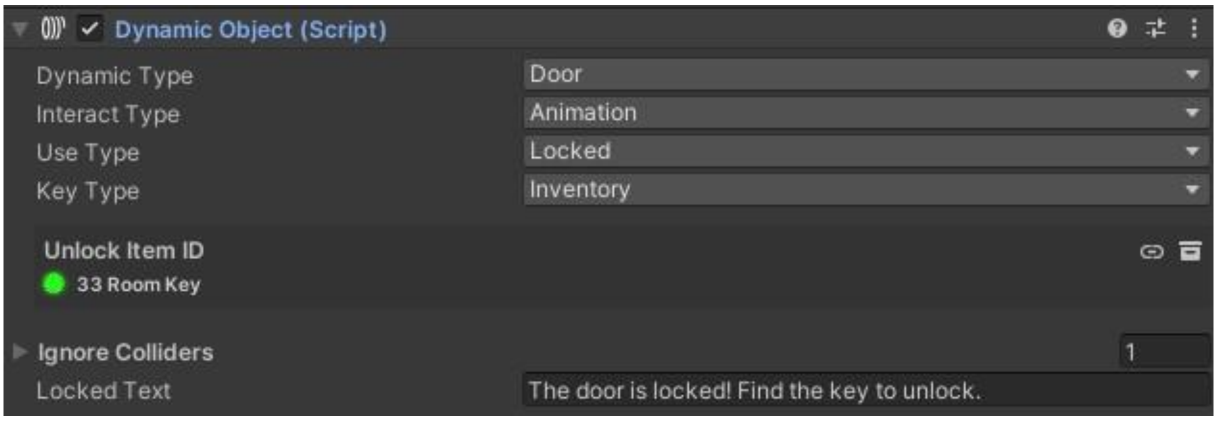


Player Footsteps
Surface ID: Texture
Surface Details: SurfaceDetails (SurfaceDetailsScriptable)

- Using surface details is very simple, just move your surface data into a script.
- To use classic surface detection, change the **Surface ID** to **Tag**.
- The **Surface Details Asset** also contains all the required functions, so obtaining elements is very easy.
- Each function of **Surface Details** is well commented.

DYNAMIC OBJECTS

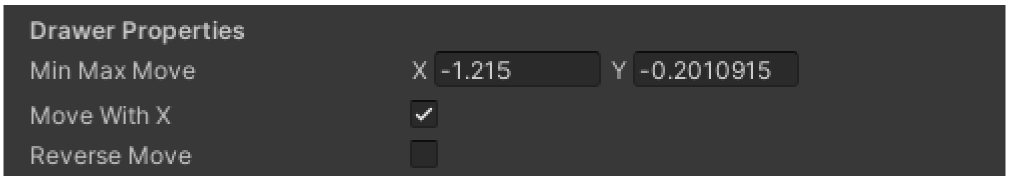
- Using a **DynamicObject.cs** script, you can easily define interactable **Doors**, **Drawers**, **Levers**, **Valves**, or **Moving Interactions**.



Dynamic Object (Script)

Dynamic Type	Door
Interact Type	Animation
Use Type	Locked
Key Type	Inventory
Unlock Item ID	33 Room Key
Ignore Colliders	1
Locked Text	The door is locked! Find the key to unlock.

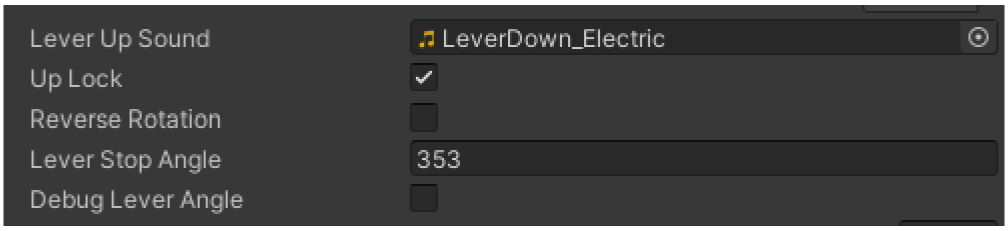
- You can define a new dynamic type by changing the **Dynamic Type** field.
- By changing the **Use Type**, you can determine whether the object is **Normal**, **Locked**, or **Jammed**.
- By changing the **Interact Type**, you can determine whether the object can be opened or closed using **Animation** or by dragging the **Mouse**.
- The **Key Type** determines how you want to unlock the **Locked Use Type**.



Drawer Properties

Min Max Move	X -1.215 Y -0.2010915
Move With X	✓
Reverse Move	

- By defining a new dynamic **Drawer** type with an interact **Mouse** type, you must assign a minimum and maximum pull position by changing the **Min Max Move** field.

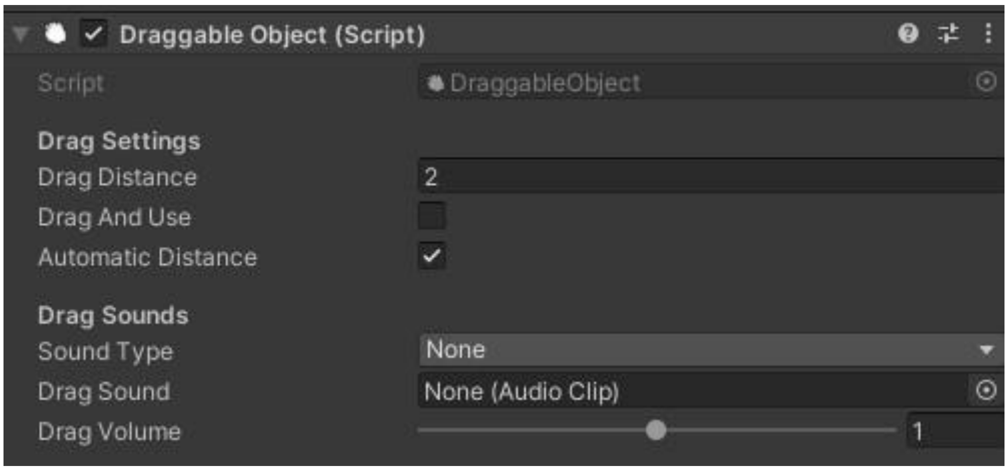


Lever Up Sound	LeverDown_Electric
Up Lock	✓
Reverse Rotation	
Lever Stop Angle	353
Debug Lever Angle	

- The same must be done with the dynamic **Lever** type by changing the **Lever Stop Angle** field. If you have trouble defining the stop angle, check the **Debug Lever Angle**.

DRAGGABLE OBJECTS

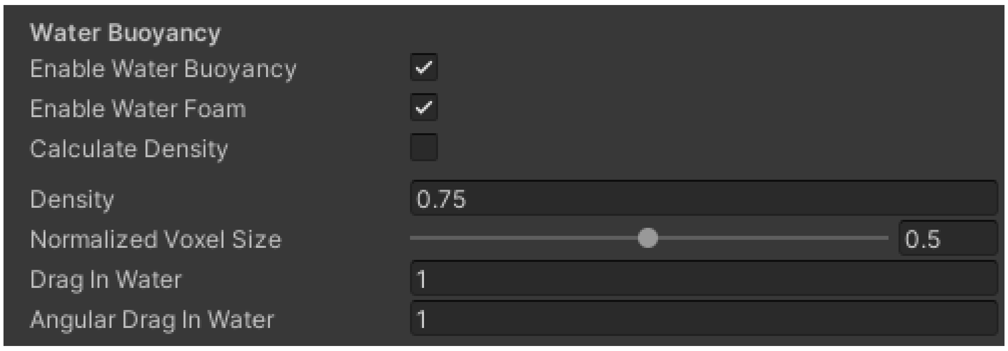
- Defining draggable objects is very simple, just change the object layer to **Interact** and add the **DraggableObject.cs** script.



Draggable Object (Script)

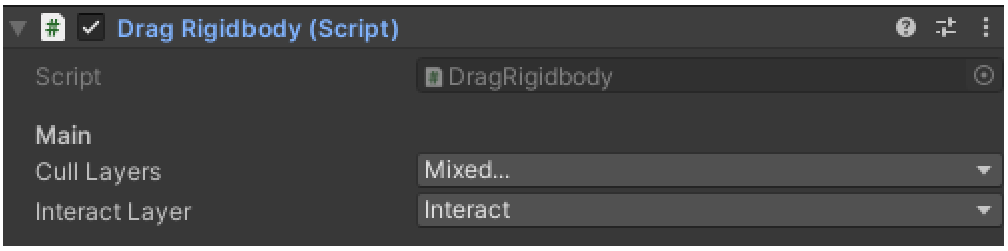
Script	DraggableObject
Drag Settings	
Drag Distance	2
Drag And Use	☐
Automatic Distance	✓
Drag Sounds	
Sound Type	None
Drag Sound	None (Audio Clip)
Drag Volume	1

- You can also determine the water behaviour of draggable objects.
- Object Density > Water Density = Lower Object Buoyancy.



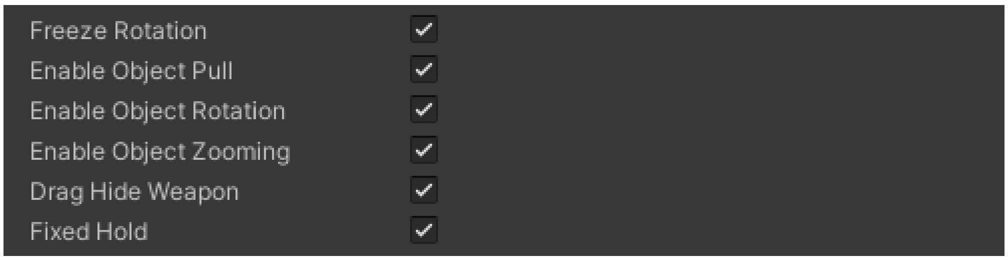
Water Buoyancy	
Enable Water Buoyancy	✓
Enable Water Foam	✓
Calculate Density	☐
Density	0.75
Normalized Voxel Size	0.5
Drag In Water	1
Angular Drag In Water	1

- You can change other settings in the **Drag Rigidbody** script located in the **PLAYER** -> **MouseLook** object.



Drag Rigidbody (Script)

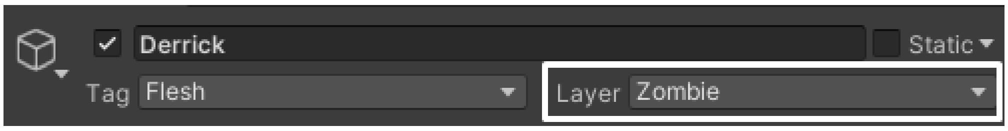
Script	DragRigidbody
Main	
Cull Layers	Mixed...
Interact Layer	Interact



Freeze Rotation	✓
Enable Object Pull	✓
Enable Object Rotation	✓
Enable Object Zooming	✓
Drag Hide Weapon	✓
Fixed Hold	✓

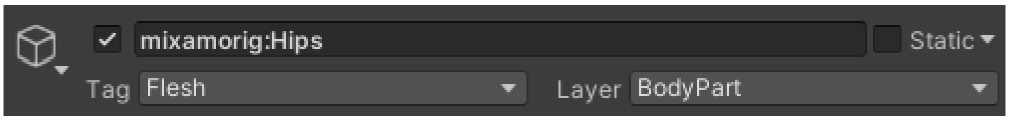
ADDING NEW ZOMBIE AI

1. Convert your **Zombie Character** to a **Ragdoll** (GameObject -> 3D Object -> Ragdoll).
2. Adjust the ragdoll colliders to match your character model.
3. Change the zombie root layer to **Zombie**.



Derrick
Static
Tag Flesh
Layer Zombie

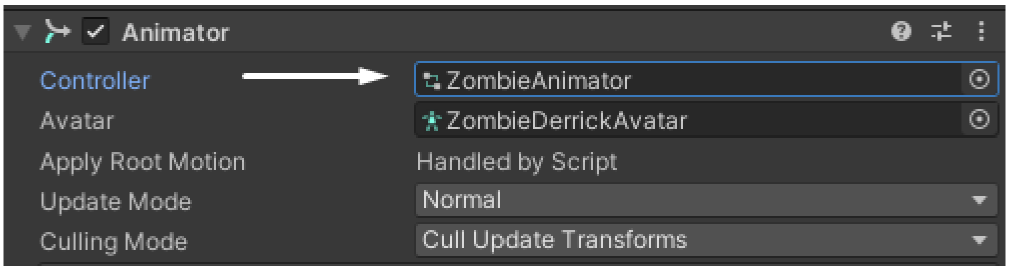
4. Then change the zombie hips layer to **BodyPart** and tag to **Flesh**.



mixamorig:Hips
Static
Tag Flesh
Layer BodyPart

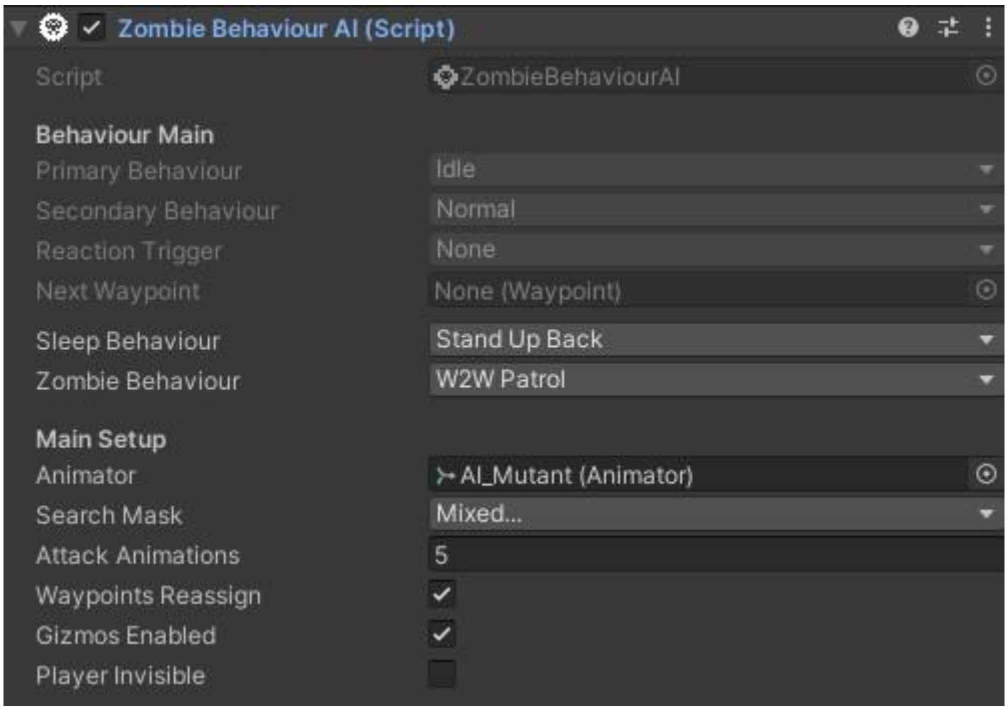
5. Assign an **Animator Controller Asset** to an **Animator** component.

- It is recommended to use the default **ZombieAnimator Controller** as it contains all required connections and behaviour scripts.



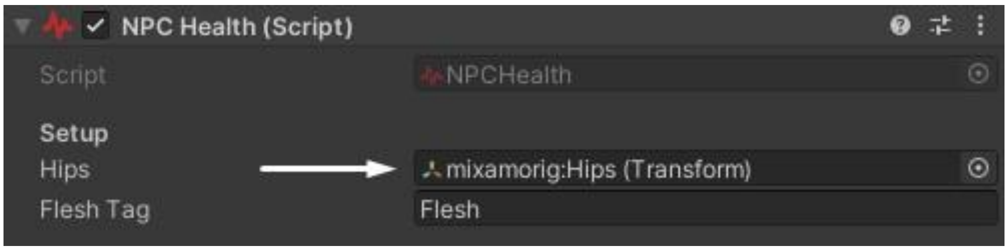
Animator
Controller ZombieAnimator
Avatar ZombieDerrickAvatar
Apply Root Motion Handled by Script
Update Mode Normal
Culling Mode Cull Update Transforms

6. Add **ZombieBehaviourAI.cs** script to zombie root.

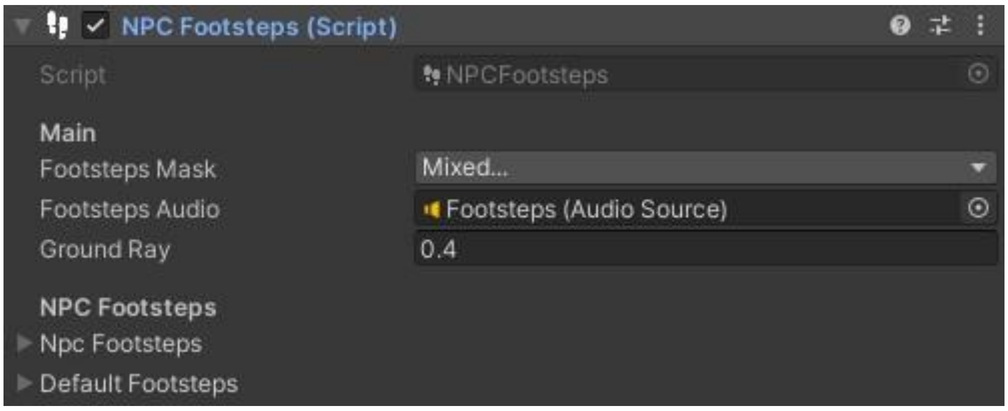


Zombie Behaviour AI (Script)
Script ZombieBehaviourAI
Behaviour Main
Primary Behaviour Idle
Secondary Behaviour Normal
Reaction Trigger None
Next Waypoint None (Waypoint)
Sleep Behaviour Stand Up Back
Zombie Behaviour W2W Patrol
Main Setup
Animator AI_Mutant (Animator)
Search Mask Mixed...
Attack Animations 5
Waypoints Reassign ✓
Gizmos Enabled ✓
Player Invisible

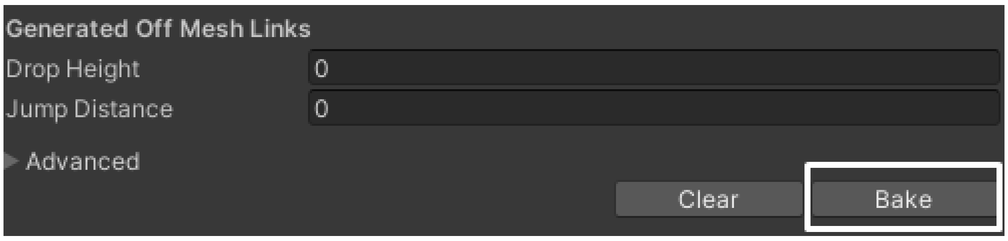
7. Add **Capsule Collider** component to the zombie root and adjust the collider to match your character model.
8. Assign **Zombie Animator** component to an **Animator** field and define a **Search Mask**.
9. Add the **NPCHealth.cs** script to the zombie root and assign zombie hips.


NPC Health (Script)
Script NPCHealth
Setup
Hips mixamorig:Hips (Transform)
Flesh Tag Flesh

10. Add the **NPCFootsteps.cs** script to the zombie root.


NPC Footsteps (Script)
Script NPCFootsteps
Main
Footsteps Mask Mixed...
Footsteps Audio Footsteps (Audio Source)
Ground Ray 0.4
NPC Footsteps
Npc Footsteps
Default Footsteps

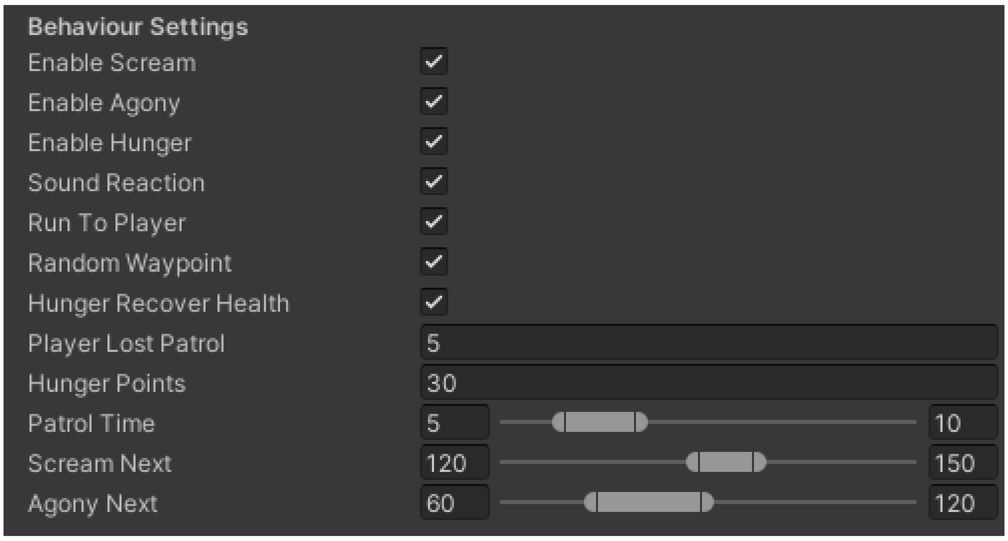
11. Create a new empty object, add an **Audio Source** component, change **Spatial Blend** to **3D** and move the object as the zombie root child object.
12. Add a **Nav Mesh Agent** component and adjust it to fit your character's model. Make sure that the **Stopping Distance** is not zero and **Auto Braking** is enabled.
13. Go to **Window -> AI -> Navigation -> Bake** and click **Bake**.


Generated Off Mesh Links
Drop Height 0
Jump Distance 0

14. Add **WaypointGroup.cs** script to the empty object.

- Waypoints are added automatically by adding empty objects to the **Waypoint Group** child object.

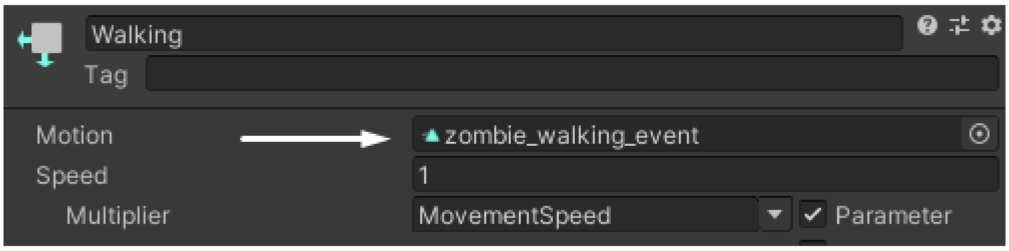
- You can define your own character behaviour by changing the **Behaviour Settings**.



Behaviour Settings

Enable Scream	✓
Enable Agony	✓
Enable Hunger	✓
Sound Reaction	✓
Run To Player	✓
Random Waypoint	✓
Hunger Recover Health	✓
Player Lost Patrol	5
Hunger Points	30
Patrol Time	5 – 10
Scream Next	120 – 150
Agony Next	60 – 120

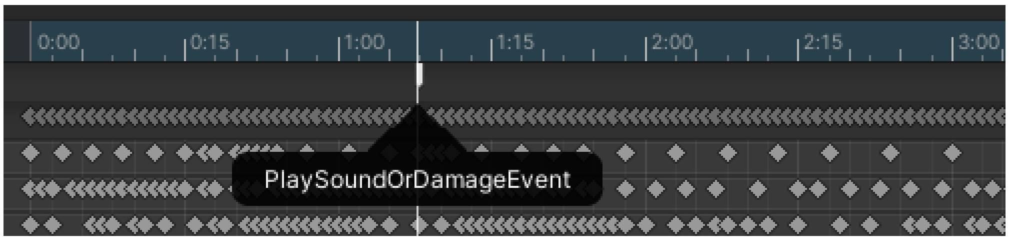
- If you want, you can also define hunger points by adding the **HungerPoint.cs** script to the object.
- Zombie will search for **Hunger Point** objects, if **Hunger** is enabled and the **Hunger Points** field in the **AI Zombie Behavior** script is zero.



Walking
Tag

Motion	zombie_walking_event
Speed	1
Multiplier	MovementSpeed ✓ Parameter

- If you have your own zombie animations, it is recommended to change the default animations of the zombie controller asset.



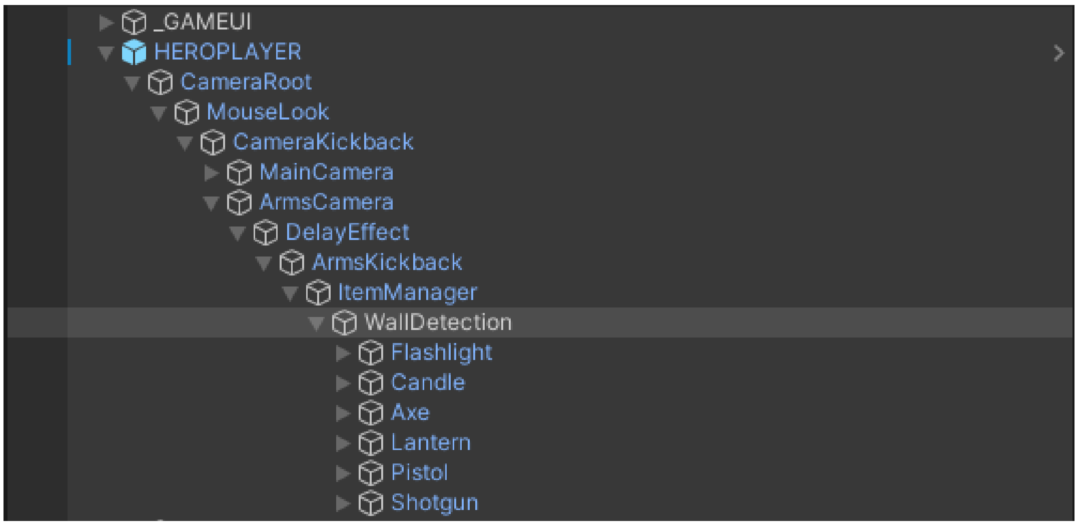
- To damage the player, you must create an animation event that calls the **PlaySoundOrDamageEvent()** function with a parameter of 0.

Sleep Behaviour: The zombie starts with a sleep animation if the player is in close distance or if it makes any impact sound, the zombie gets up and changes his sleep behavior to None.

Zombie Behaviour: The basic awake behavior of the zombie. Waypoint to Waypoint, Waypoint to Waypoint with Patrol, Waypoint to Waypoint with Idle.

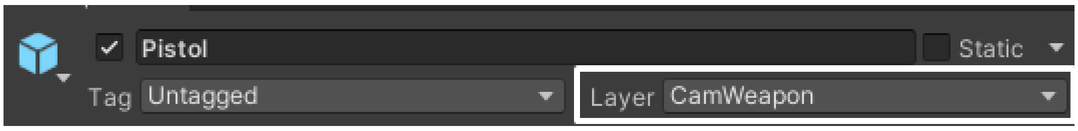
ADDING NEW PLAYER WEAPONS

1. Locate the **WallDetection** object in the **PLAYER** object, which contains all the usable objects.



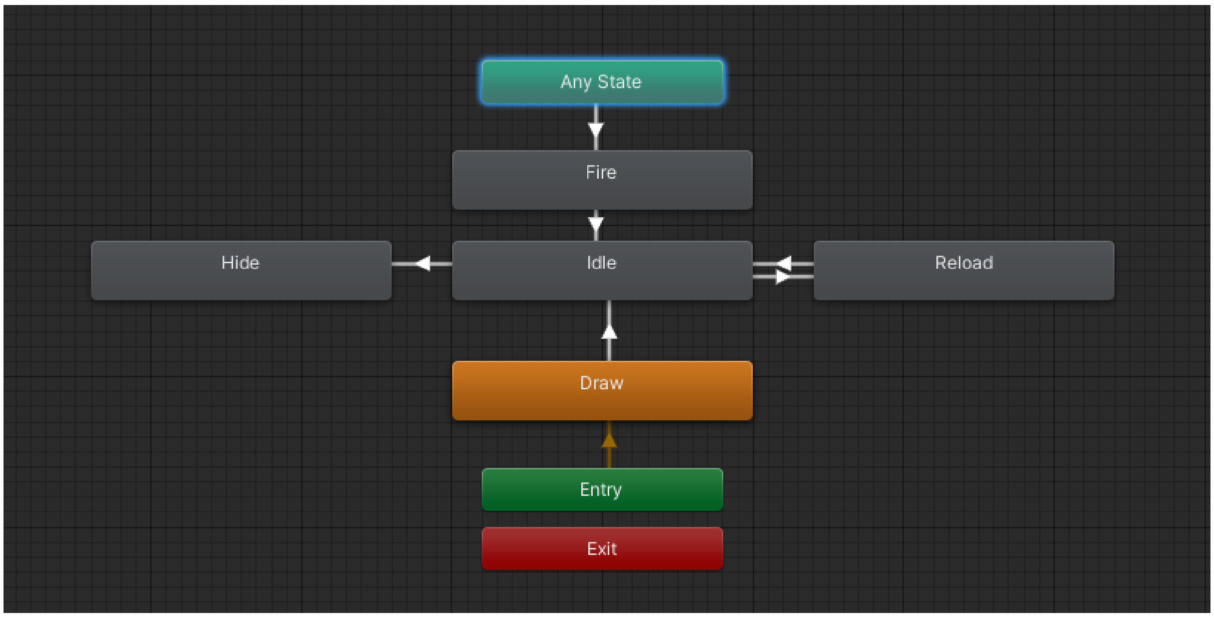
```
▶ _GAMEUI
▼ HEROPLAYER
  ▼ CameraRoot
    ▼ MouseLook
      ▼ CameraKickback
        ▶ MainCamera
        ▼ ArmsCamera
          ▼ DelayEffect
            ▼ ArmsKickback
              ▼ ItemManager
                ▼ WallDetection
                  ▶ Flashlight
                  ▶ Candle
                  ▶ Axe
                  ▶ Lantern
                  ▶ Pistol
                  ▶ Shotgun
```

2. Duplicate one of these items and replace the disabled object inside the duplicated object with your object.
3. Set a new layer of object to the **CamWeapon**.

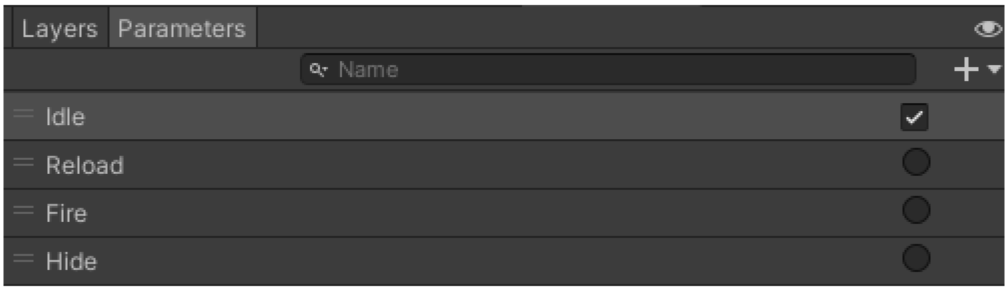


✓ Pistol — Static
Tag: Untagged — Layer: CamWeapon

4. Create a weapon **Animator** asset.

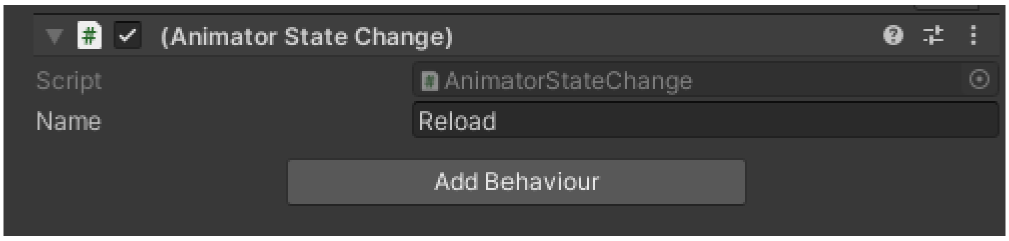


5. Add the required parameters: **(bool) Idle = true**, **(trigger) Reload**, **Shoot**, **Hide**



Idle	✓
Reload	●
Fire	●
Hide	●

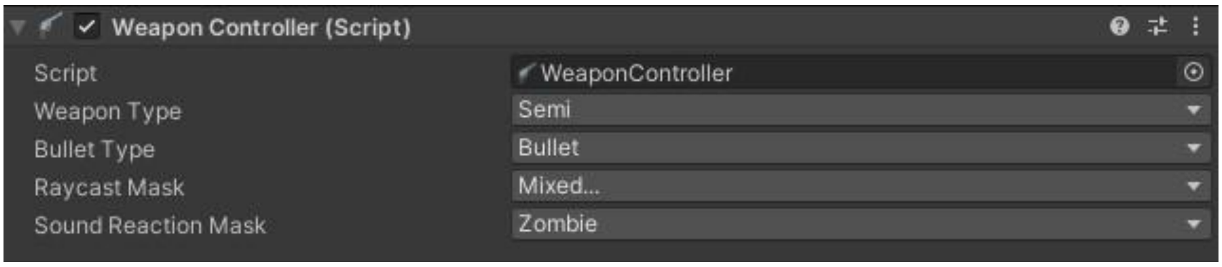
6. Create all required state transitions.
7. Add **AnimatorStateChange** behaviour to **Draw, Hide, Reload** states with the name **Draw, Hide, Reload** that the **WeaponController** script uses.



Script	AnimatorStateChange
Name	Reload

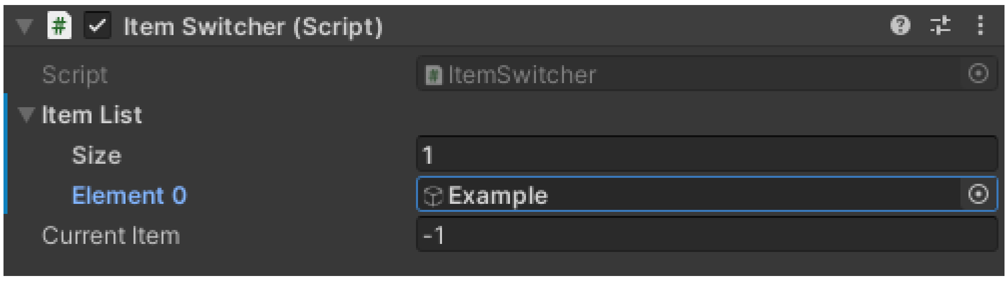
- This behavior script finds and sends state data to the **WeaponController** script using the **IOnAnimatorState** interface.
- It is important to write exactly **Draw, Hide or Reload** in the name field, depending on the state.

8. Add the **WeaponController.cs** script to the weapon root object and set the weapon parameters.



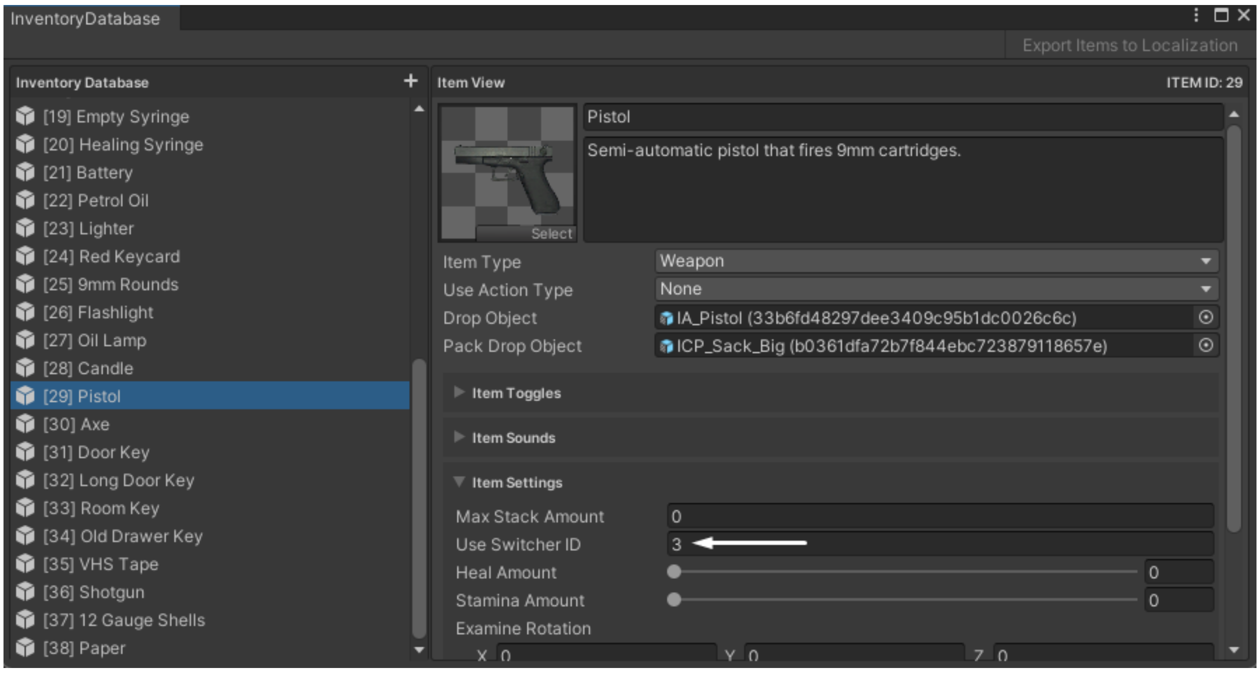
Script	WeaponController
Weapon Type	Semi
Bullet Type	Bullet
Raycast Mask	Mixed...
Sound Reaction Mask	Zombie

9. Go to the **ItemManager** object and add a new weapon to the **ItemSwitcher -> Item List**.



Script	ItemSwitcher
Item List	
Size	1
Element 0	Example
Current Item	-1

10. To use your weapon from the inventory, you must define the **Use Switcher ID** field from the **Item Settings** drop-down menu. **Weapon ID** is the ID of the **Item List** element.



InventoryDatabase

Export Items to Localization

Inventory Database

- [19] Empty Syringe
- [20] Healing Syringe
- [21] Battery
- [22] Petrol Oil
- [23] Lighter
- [24] Red Keycard
- [25] 9mm Rounds
- [26] Flashlight
- [27] Oil Lamp
- [28] Candle
- [29] Pistol
- [30] Axe
- [31] Door Key
- [32] Long Door Key
- [33] Room Key
- [34] Old Drawer Key
- [35] VHS Tape
- [36] Shotgun
- [37] 12 Gauge Shells
- [38] Paper

Item View — ITEM ID: 29

Pistol

Semi-automatic pistol that fires 9mm cartridges.

Item Type: Weapon
Use Action Type: None
Drop Object: IA_Pistol (33b6fd48297dee3409c95b1dc0026c6c)
Pack Drop Object: ICP_Sack_Big (b0361dfa72b7f844ebc723879118657e)

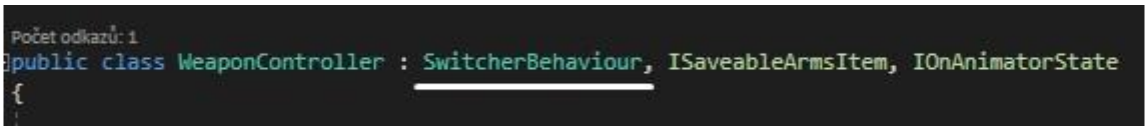
Item Toggles

Item Sounds

Item Settings

Max Stack Amount: 0
Use Switcher ID: 3
Heal Amount: 0
Stamina Amount: 0
Examine Rotation: X 0 Y 0 Z 0

- You can also define your own usable weapons/items by adding the **SwitcherBehaviour** subclass to your own script.



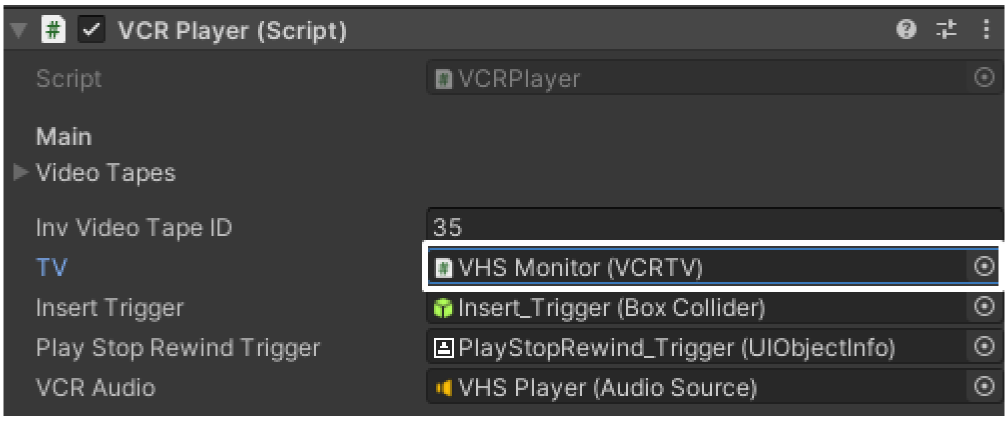
```
Počet odkazů: 1
public class WeaponController : SwitcherBehaviour, ISaveableArmsItem, IOnAnimatorState
{
```

- You can add an **ISaveableArmsItem** interface to a script to save and load usable object data.

SETTING UP VHS PLAYER

- Supported video formats: [Video file compatibility]

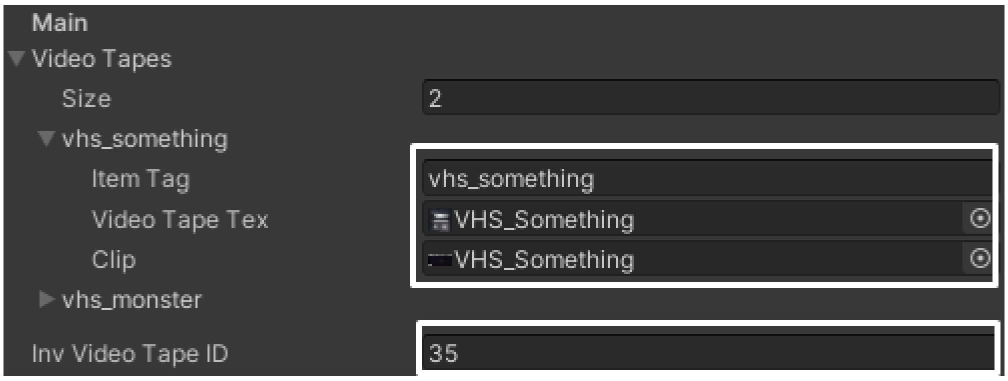
1. Move **VHS Player** and **VHS Monitor** from **Resources** folder to the scene.
2. Connect **VHS Player** with **VHS Monitor**.



VCR Player (Script)

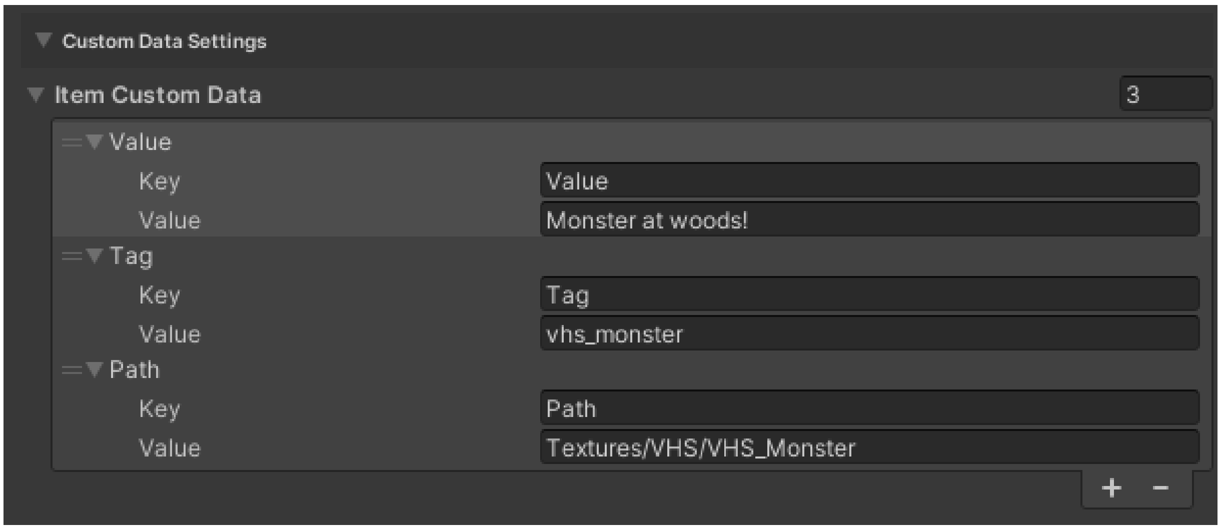
Script	VCRPlayer
Main	
Video Tapes	
Inv Video Tape ID	35
TV	VHS Monitor (VCRTV)
Insert Trigger	Insert_Trigger (Box Collider)
Play Stop Rewind Trigger	PlayStopRewind_Trigger (UIObjectInfo)
VCR Audio	VHS Player (Audio Source)

3. Set **Inventory Video Tape ID** and add your video tapes.



Main	
Video Tapes	
Size	2
vhs_something	
Item Tag	vhs_something
Video Tape Tex	VHS_Something
Clip	VHS_Something
vhs_monster	
Inv Video Tape ID	35

4. In **Interactive Item** script, set VHS Tape **Item Custom Data**.



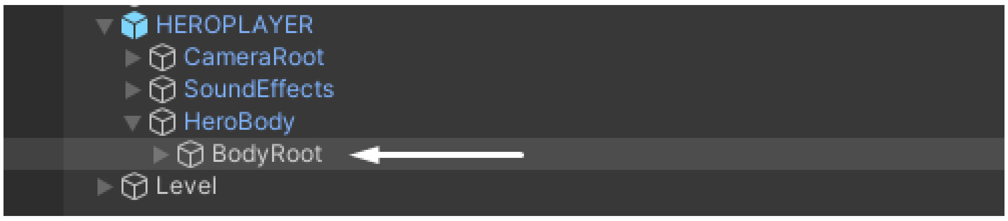
Custom Data Settings

Item Custom Data	3
Value	
Key	Value
Value	Monster at woods!
Tag	
Key	Tag
Value	vhs_monster
Path	
Key	Path
Value	Textures/VHS/VHS_Monster

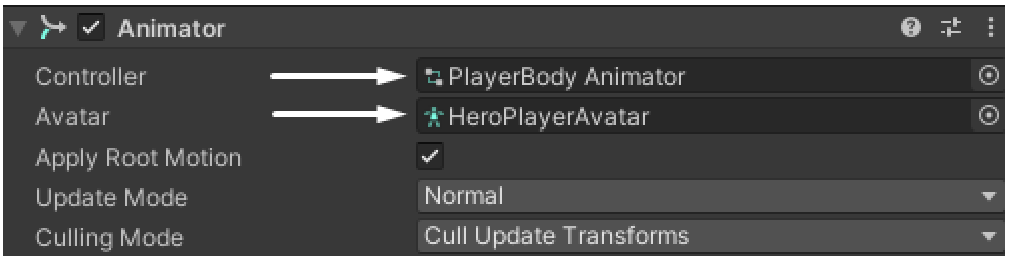
- The **Path** value is the path inside the **Resources** folder. All keys must be same!

ADDING CUSTOM PLAYER MODEL

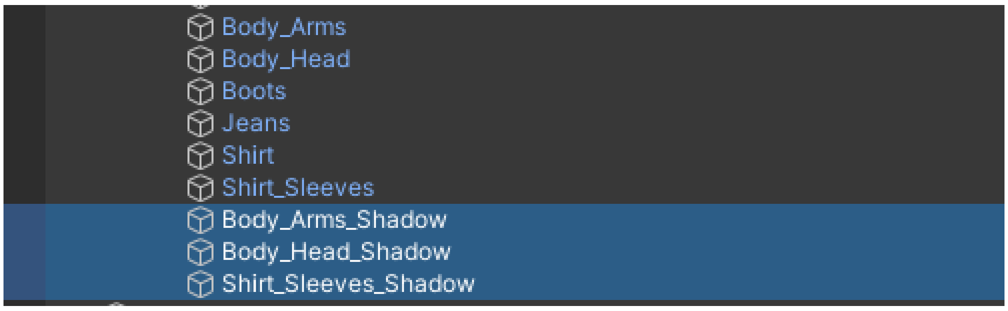
1. Replace the **BodyRoot** object with your **Player** model.

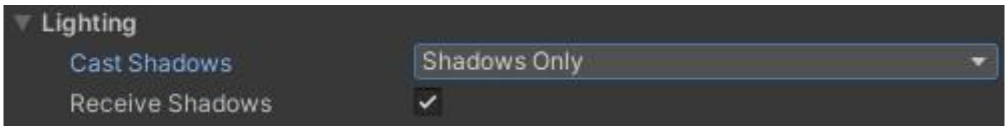

HEROPLAYER
CameraRoot
SoundEffects
HeroBody
BodyRoot
Level

2. Assign the **PlayerBody Animator** asset to the **Controller** field and make sure you have the **Humanoid Avatar** assigned to the **Avatar** field.

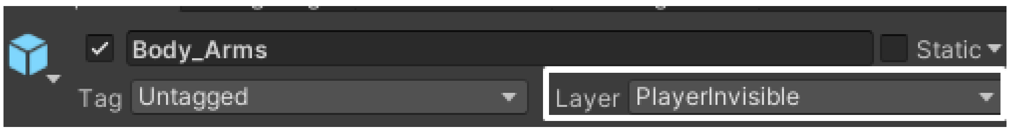

Animator
Controller: PlayerBody Animator
Avatar: HeroPlayerAvatar
Apply Root Motion: ✓
Update Mode: Normal
Culling Mode: Cull Update Transforms

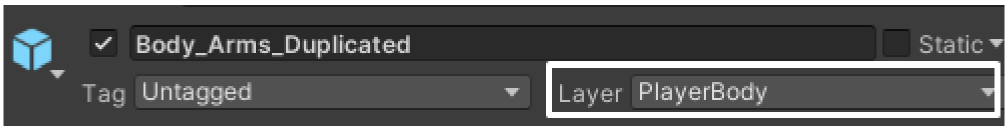
3. If you want your hands and head to be invisible to the **Main Camera**, you must duplicate these objects and change their **Cast Shadows** field to **Shadows Only** in the **Lighting** drop-down menu.


Body_Arms
Body_Head
Boots
Jeans
Shirt
Shirt_Sleeves
Body_Arms_Shadow
Body_Head_Shadow
Shirt_Sleeves_Shadow

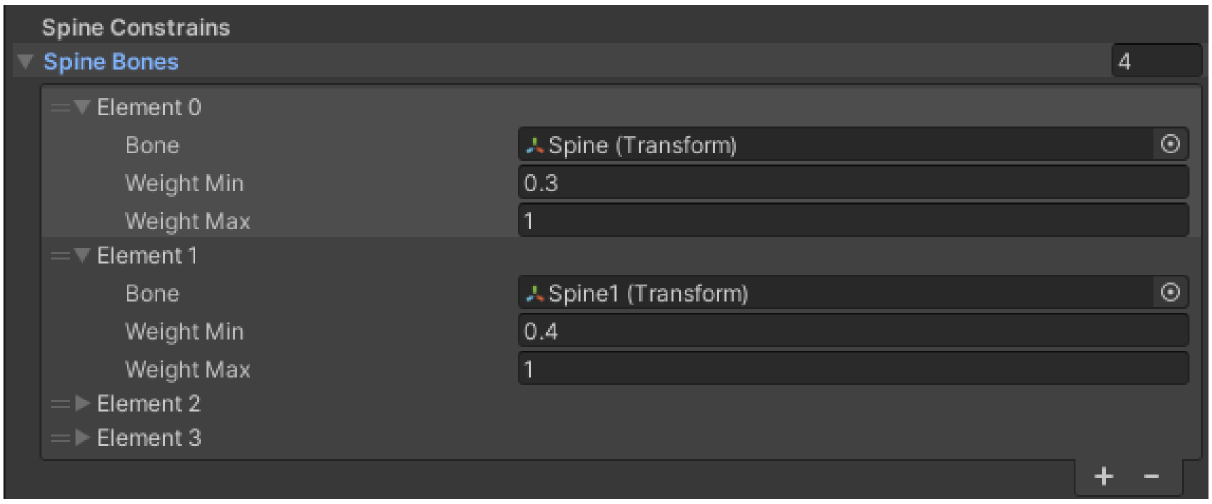

Lighting
Cast Shadows: Shadows Only
Receive Shadows: ✓

4. Next, you need to change the layer of the original duplicated objects to **PlayerInvisible** and the other player objects to **PlayerBody**.


Body_Arms
Static
Tag: Untagged
Layer: PlayerInvisible

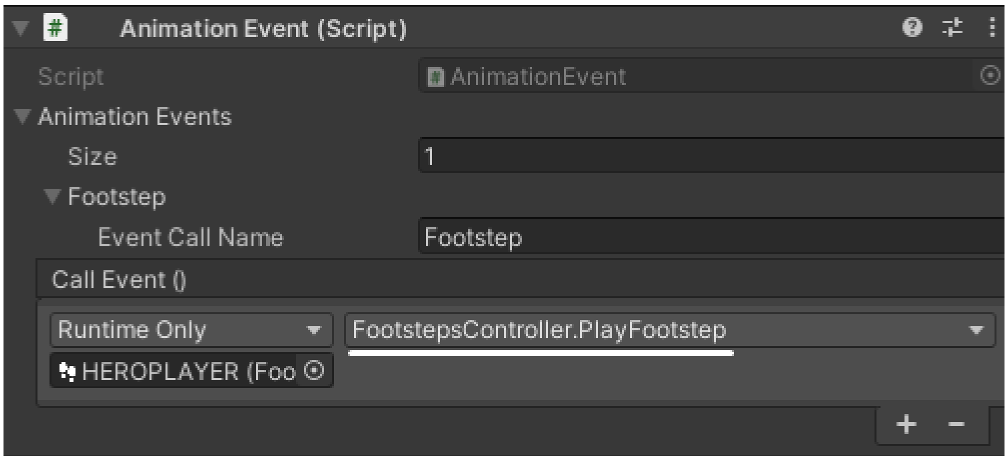

Body_Arms_Duplicated
Static
Tag: Untagged
Layer: PlayerBody

5. Assign a player's Spine Bones in the **Body Animator**.



Spine Constrains
Spine Bones 4
- Element 0
  - Bone: Spine (Transform)
  - Weight Min: 0.3
  - Weight Max: 1
- Element 1
  - Bone: Spine1 (Transform)
  - Weight Min: 0.4
  - Weight Max: 1
- Element 2
- Element 3

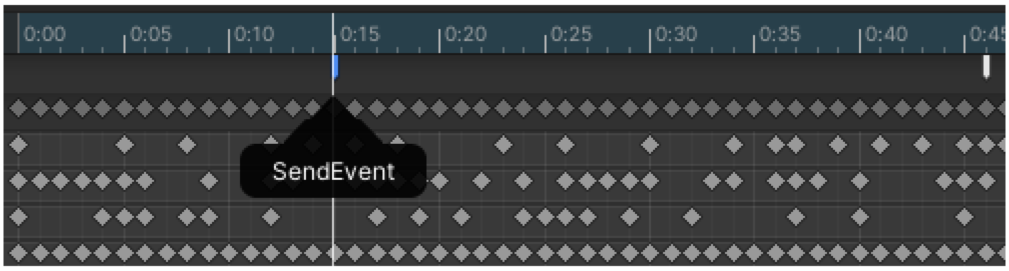
6. To perform event-based footsteps, you must add the **AnimationEvent.cs** script to the **HeroBody -> HeroRoot** object and create a **FootstepsController -> PlayFootstep()** event.



Animation Event (Script)
Script: AnimationEvent
Animation Events
- Size: 1
- Footstep
  - Event Call Name: Footstep

Call Event ()
Runtime Only | FootstepsController.PlayFootstep
HEROPLAYER (Foo

7. Then, at a specific animation time, create an event that points to the **AnimationEvent -> SendEvent()** function.



8. As the **SendEvent()** parameter set the text of the **Event Call Name**.

EMERALD AI 2.0 INTEGRATION

1. Open an empty scene and import **Emerald AI** first and then **Horror FPS KIT**.
2. Go to **Horror FPS KIT\HFPS Assets\Content\Scripts\Core\Examples** and unzip the **EmeraldAI.zip** package.
3. Replace default **EmeraldAIPlayerDamage.cs** script with an extracted version.If the script already contains the **DamageHFPSPlayer()** function, uncomment itand skip this step.
4. Move **EmeralAISendDamage.cs** script right after the enemy **EmeraldAISystem** script.
5. Follow the **Emerald AI** instructions and create enemies.

HDRP/URP PARTIALL SUPPORT

- The **Horror FPS KIT** has partial support for **HDRP** or **URP,** which means that the kit is compatible with these rendering pipelines, but after setting up on these pipelines, pink materials or post-processing problems may occur.

- If you choose to use the **URP** rendering pipeline, some scenes may contain pink materials. This problem can be easily solved by changing all shaders of pink materials to **URP** default shaders. The reason you need to do this is because Unity has chosen not to support other shaders, such as **URP** or **HDRP** shaders.

- If you choose to use the **HDRP** rendering pipeline, you will experience the same problem with pink materials as in **URP**, but there will also be an incompatibility with post-processing. This is because the **HDRP** rendering pipeline uses its own post-processing system and the camera **blur effect** will not be compatible anymore. You can also easily solve the problem with the pink materials, but if you want to use the **blur effect**, you have to write a brand new **shader** and **script**, or you can use another method instead of the blur effect, such as a transparent black panel or something else.

CREDITS

- This kit was developed and designed by **ThunderWire Studio**

  © All rights reserved.
- Almost all assets are created by **ThunderWire Studio**, except those that are downloaded under a royalty free license.

ASSET CREDITS

1. 4 Industrial barrels by Vertex Field
https://assetstore.unity.com/packages/3d/props/4-industrial-barrels-76538

2. PBR - Hospital Horror Pack. Free by DNK_DEV
https://assetstore.unity.com/packages/3d/environments/pbr-hospital-horror-pack-free-80117

3. Retro Lamps v.1 by Artur G.
https://assetstore.unity.com/packages/3d/props/interior/retro-lamps-v-1-19601

4. Rusty Pack Models by Karol Led
https://assetstore.unity.com/packages/3d/props/rusty-pack-models-14412

5. Books by VIS Games
https://assetstore.unity.com/packages/3d/props/interior/books-3356

6. Kitchen Props Free by Jake Sullivan
https://assetstore.unity.com/packages/3d/props/interior/kitchen-props-free-80208

7. Animated Old Chest by NOT_Lonely
https://assetstore.unity.com/packages/3d/props/animated-old-chest-20179

BUG, ERROR REPORT

- If you find any issues with our kit, join to our **Discord** server, verify your purchase by following the instructions on the **#become-verified** channel, and send a support message to the appropriate channel.

- For website customers, log in to your account, click on the drop-down menu in the top right corner and click on **Customer Support**.

USEFUL LINKS

- **ThunderWire Studio** - Youtube Channel

- **ThunderWire Studio** - Developer Website

- **ThunderWire Studio** - AssetStore

- **Horror FPS KIT** – HFPS Website

- **Horror FPS KIT** – HFPS AssetStore