GAME KIT CONTROLLER 2.4d

By Two Cubes

Table of content

INTRODUCTION

Thanks for your purchase and interest in the asset, you won't regret it. The main purpose of this asset is to learn every aspect of a videogame and at the same time, make a solid controller, so anyone can make a quick prototype and try any idea, focusing in the game itself rather that spend time in systems or components.

Also, this asset will be updated forever or until it is perfect and allows to create literally any type of game, what happen first.

WARNING: THIS DOCUMENTATION IS NOT COMPLETE FOR THE LAST ASSET VERSION. BUT IS CURRENTLY IN PROCESS. EVERY BULLET ELEMENT BELOW IS TO SHOW HOW THE ASSET ORGANIZES AND ALL OF THEM WILL BE EXPLAINED IN EVERY DOC UPDATE.

FOR NOW, THIS IS THE LASTEST VERSION OF THIS FILE, WHICH WILL INCLUDE TUTORIALS FOR EVERY SYSTEM IN THE ASSET.

THIS FILE WILL BE UPDATED IN THE FORUM OF GKC EVERY WEEK UNTIL IS TOTALLY COMPLETE. YOU CAN FIND THE FORUM HERE:

https://forum.unity.com/threads/game-kit-controler-1st-3rd-controller-with-weapons-vehicles-2-4-released.351456/

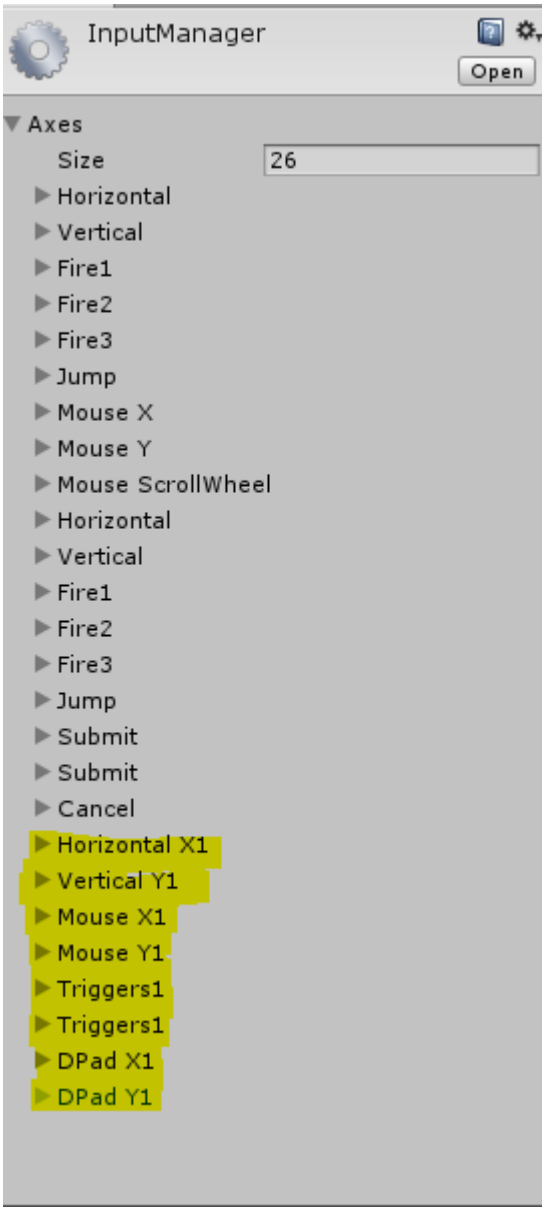
OF COURSE, IN EVERY UPDATE THE LAST VERSION OF THIS DOC WILL BE INCLUDED.

FIRST RUN

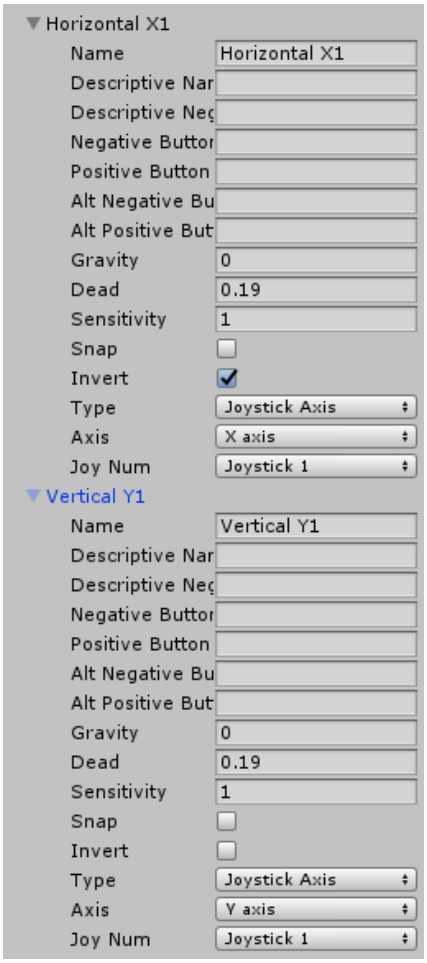
You can import this asset into an empty project or a project with previous content. If you import the asset into an existing project, you can uncheck the option to import the project settings, but in this case, you have to do the next (else, jump to First steps and basic use):

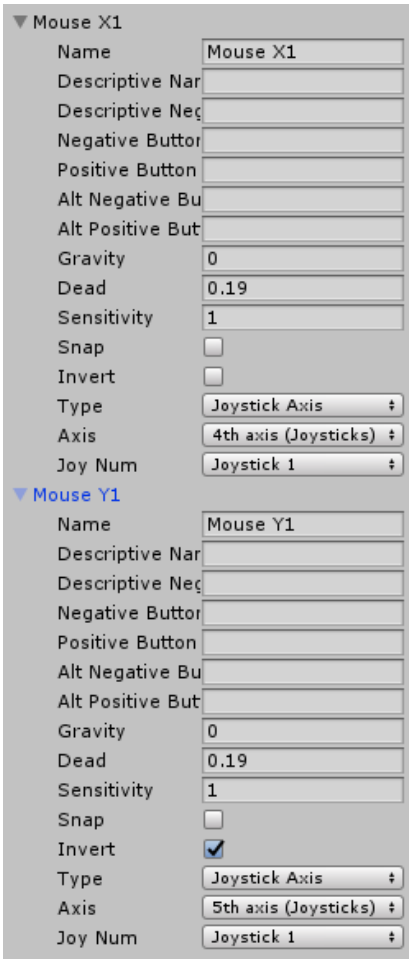
ADD AXES IN THE INPUT MANAGER

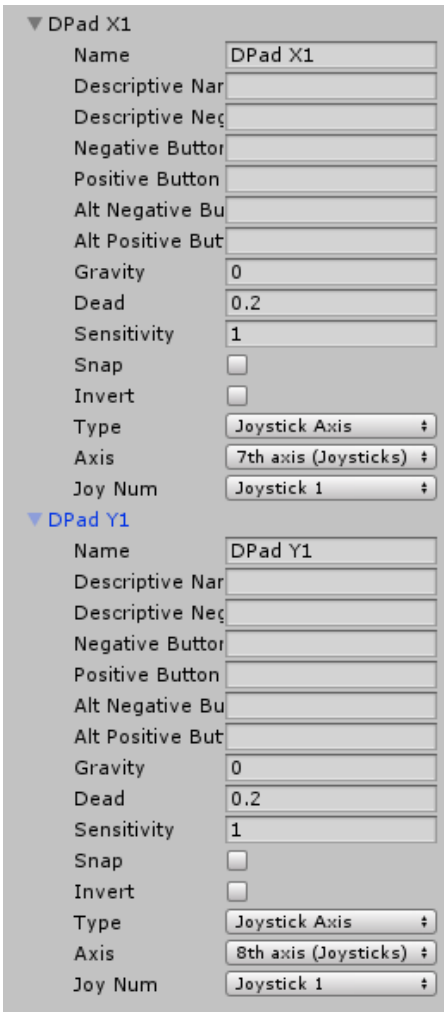
If you import the asset into a project without the Project.Settings, you have to add these axes (inside the yellow color) into the InputManager of unity if you are going to use a gamepad, in Edit->Preferences->Input. These axes are used for the gamepad, but it needs to be configured if the settings of the asset are not imported (This step won't be necessary in the next update).

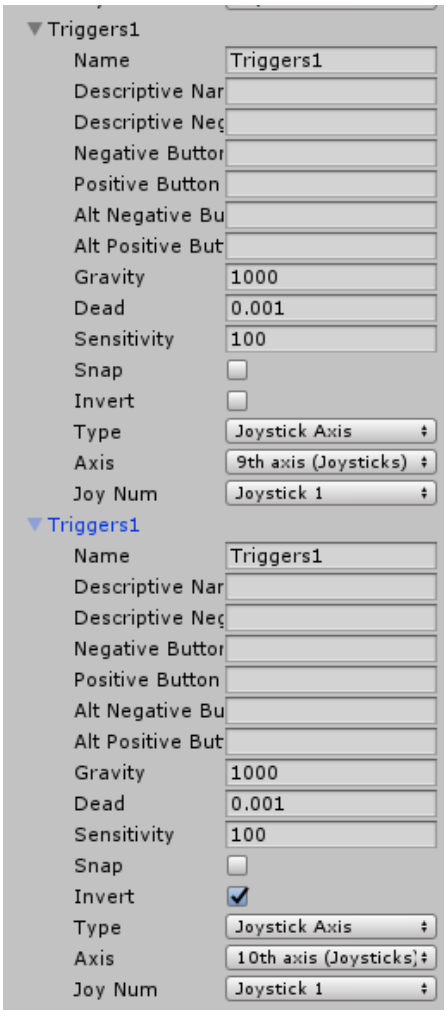


The values to configure are the next:





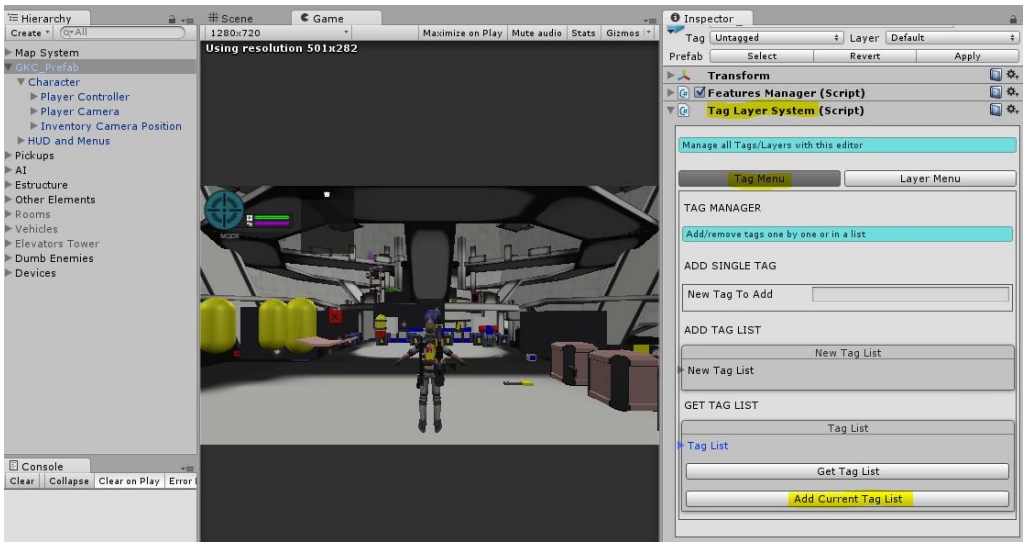

DPad X1
Name
DPad X1
Descriptive Nar
Descriptive Neg
Negative Buttor
Positive Button
Alt Negative Bu
Alt Positive But
Gravity
0
Dead
0.2
Sensitivity
1
Snap
Invert
Type
Joystick Axis
Axis
7th axis (Joysticks)
Joy Num
Joystick 1
DPad Y1
Name
DPad Y1
Descriptive Nar
Descriptive Neg
Negative Buttor
Positive Button
Alt Negative Bu
Alt Positive But
Gravity
0
Dead
0.2
Sensitivity
1
Snap
Invert
Type
Joystick Axis
Axis
8th axis (Joysticks)
Joy Num
Joystick 1


Triggers1
Name
Triggers1
Descriptive Nar
Descriptive Neg
Negative Buttor
Positive Button
Alt Negative Bu
Alt Positive But
Gravity
1000
Dead
0.001
Sensitivity
100
Snap
Invert
Type
Joystick Axis
Axis
9th axis (Joysticks)
Joy Num
Joystick 1
Triggers1
Name
Triggers1
Descriptive Nar
Descriptive Neg
Negative Buttor
Positive Button
Alt Negative Bu
Alt Positive But
Gravity
1000
Dead
0.001
Sensitivity
100
Snap
Invert
Type
Joystick Axis
Axis
10th axis (Joysticks)
Joy Num
Joystick 1

ADD THE TAGS AND LAYERS

In case you import the asset to a project without the Project.Settings, you need to add the tags and layers used in this asset. To do this automatically, you need to go the demo scene in the asset, or drag and drop the GKC_Prefab. Then, go to the inspector of that gameObject and go to Tag Layer System inspector.

Open the Tag Menu and press the button Add Current Tag List.

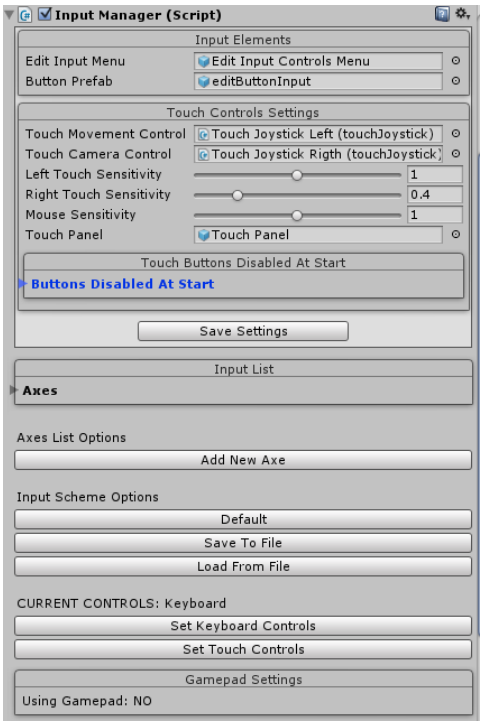


Do the same with the layers, pressing Layer Menu and the Add Current Layer List. After that, all the tags and layer used in the asset are assigned correctly to every object and prefab of the asset.

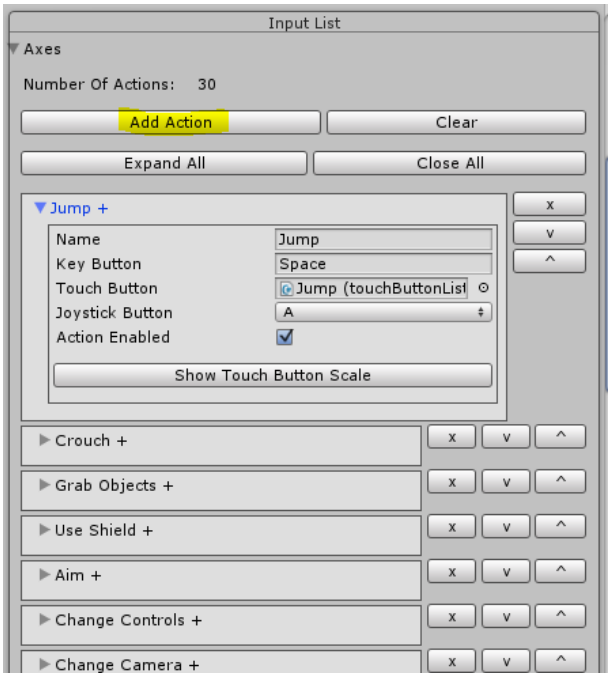
FIRST STEPS AND BASIC USE

INPUT MANAGER

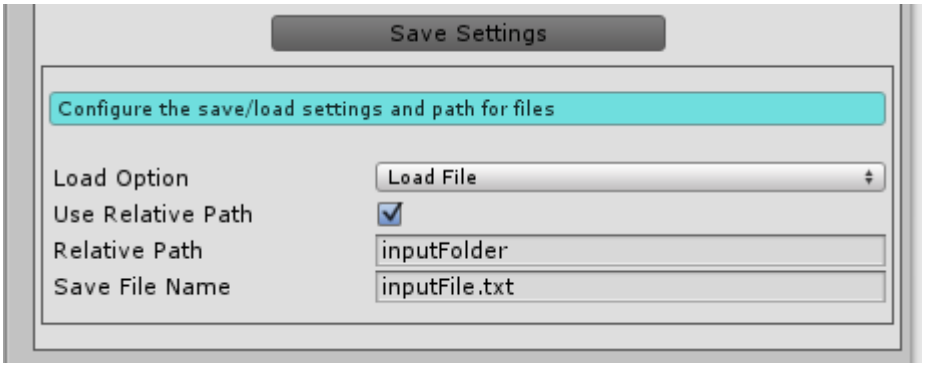
You can check all the actions in the pause menu by pressing Esc, then select Edit Control Input. In that window you can see all the actions defined in the Input Manager. This component is assigned in the gameObject “Character”.



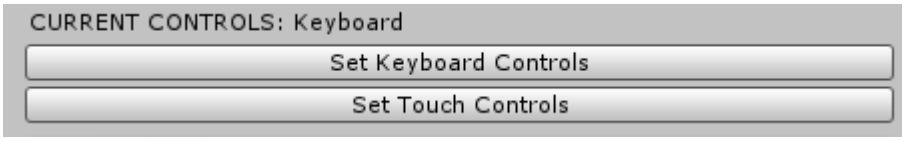
In the inspector you can add, remove, enable and disable every action and assign a keyboard button, a gamepad button and a touch button.



Also, you can edit the path to save the current map button, by pressing the button Save Settings. This file will be stored in the persistent path by default or in a relative path to your project folder if you enable the option “Use Relative Path”. IMPORTANT: if you are making a game for mobile, set the option Use Relative Path to false.



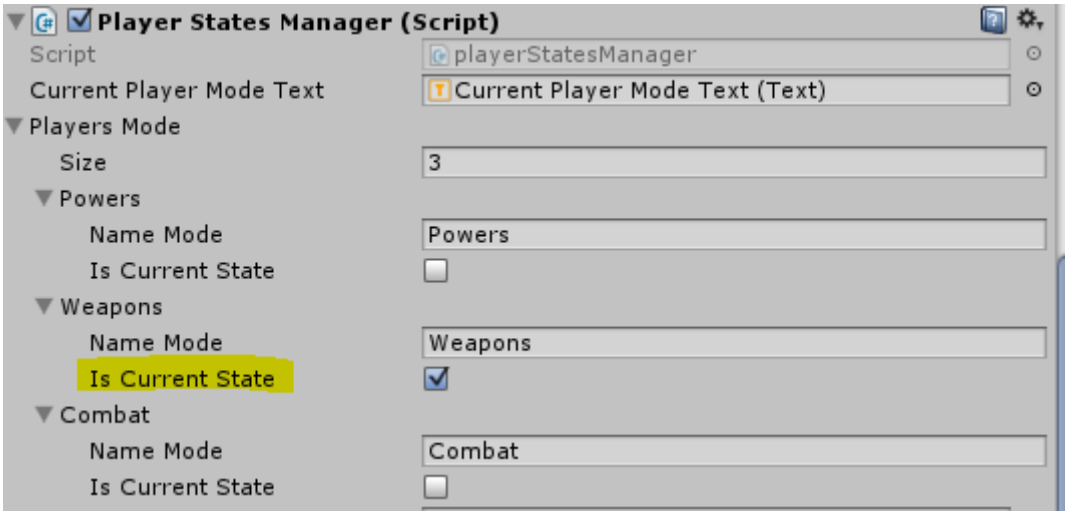
In the options below, you can select the keyboard or touch controls as input when the game starts. Also, in game, you can switch between these two modes with “C” by default if you are in keyboard controls. If you are in this touch controls mode, press Esc and then press “Switch Control Type”. This option can be used in both modes. The label Current Controls shows what current control scheme is currently selected.



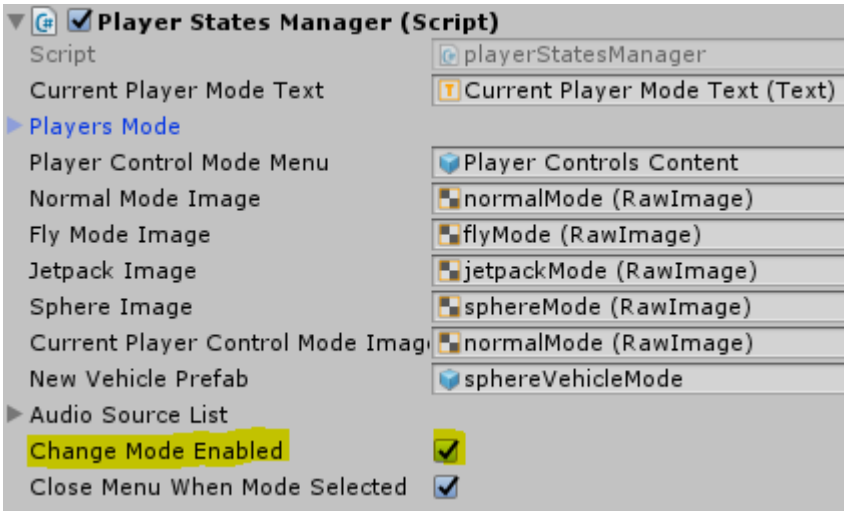
PLAYER CONTROL MODE

The player has 3 modes: weapons, powers and combat. To change between them, press H (by default). You can see the current state by checking the name in the upper left corner of the screen, below the health and energy bar.

The weapon mode is the default mode. In Player States Manager (in the gameObject Player Controller) you can choose the default mode at the beginning of the game.



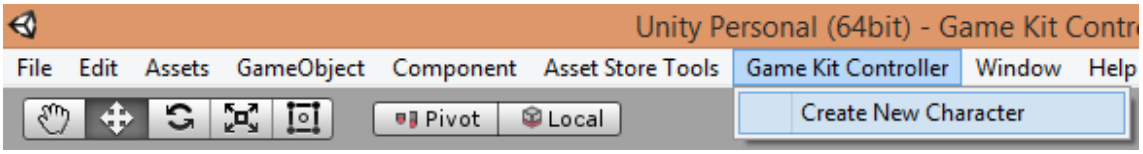
Also, you can configure if the player can change between these modes or not, so the game can always has the same mode or all of them. This option is called, Change Mode Enabled.



CREATE A NEW CHARACTER

To set your model, follow these steps:

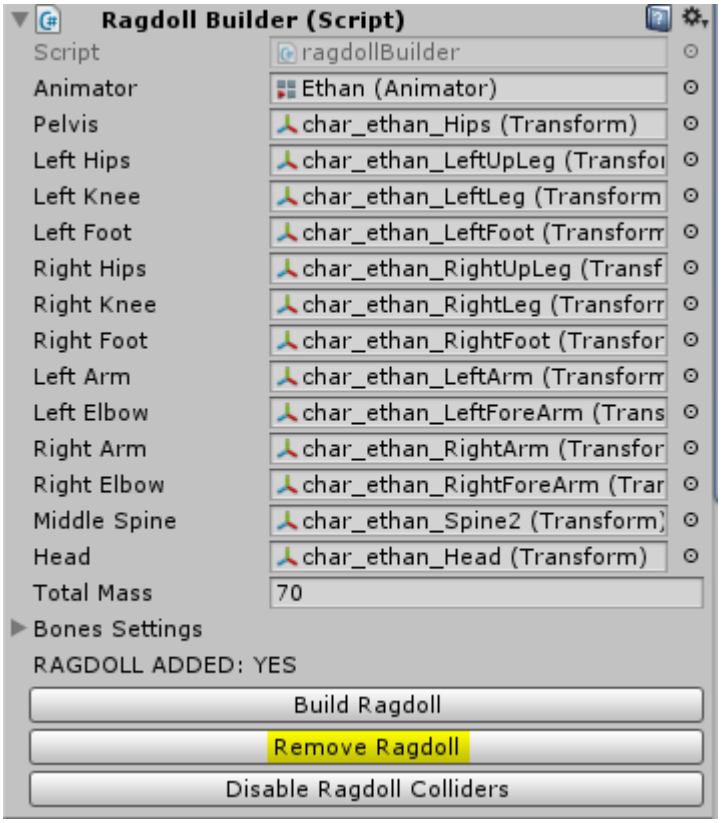
- In the tools bar press Game Kit Controller and Create New Character.



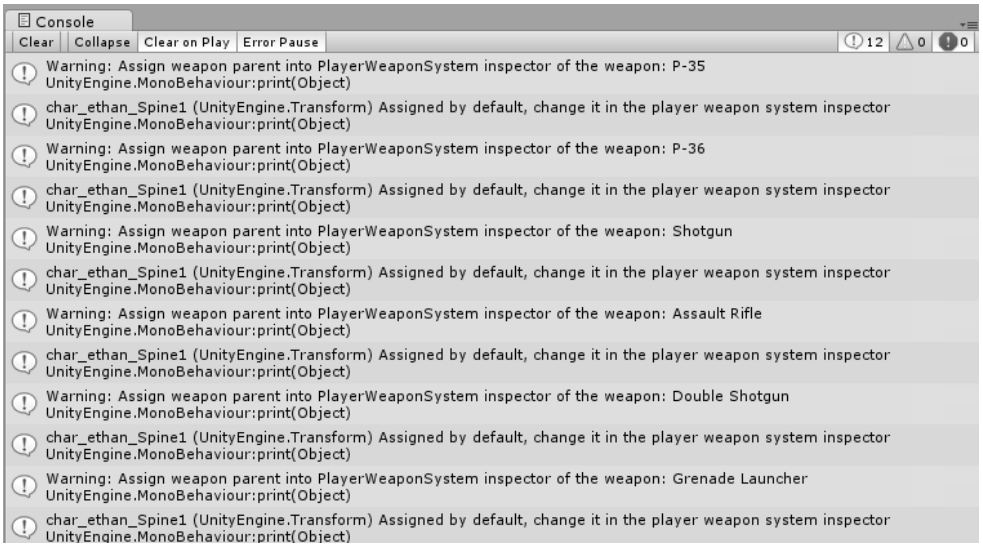
- In that window select a humanoid model and press the Create Player button.

- The new player will be place in the scene, in the position where the editor camera is looking.
- This process creates the ragdoll of the new player automatically. Here you can see how it looks.

- This ragdoll can be configured or removed. For this go to Ragdoll Builder and press the button Remove Ragdoll. You can also configure its scale and mass in the option Bone Settings.



- Then you will see the messages in the console about the weapons.



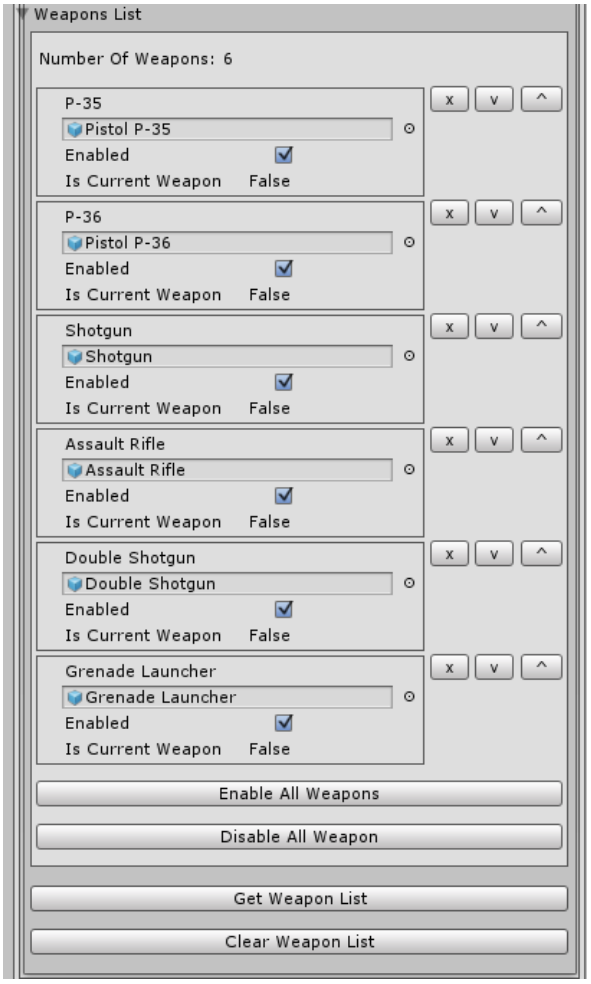
- These messages is to tell the user to configure the correct weapon parent, which is the place where every weapon is parented while is not being used, like the back, the leg, etc… By default, the weapon is placed in the back. To configure a better parent check the next steps.

SET THE DEFAULT WEAPONS IN THE NEW MODEL

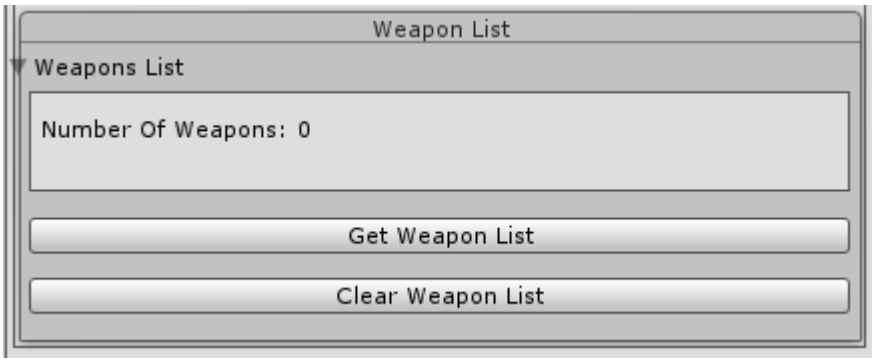
To manage the player's weapons, go to Player Controller gameObject, and to the inspector Player Weapons Manager.

IF YOU DON'T NEED TO USE THE WEAPONS, follow these steps:

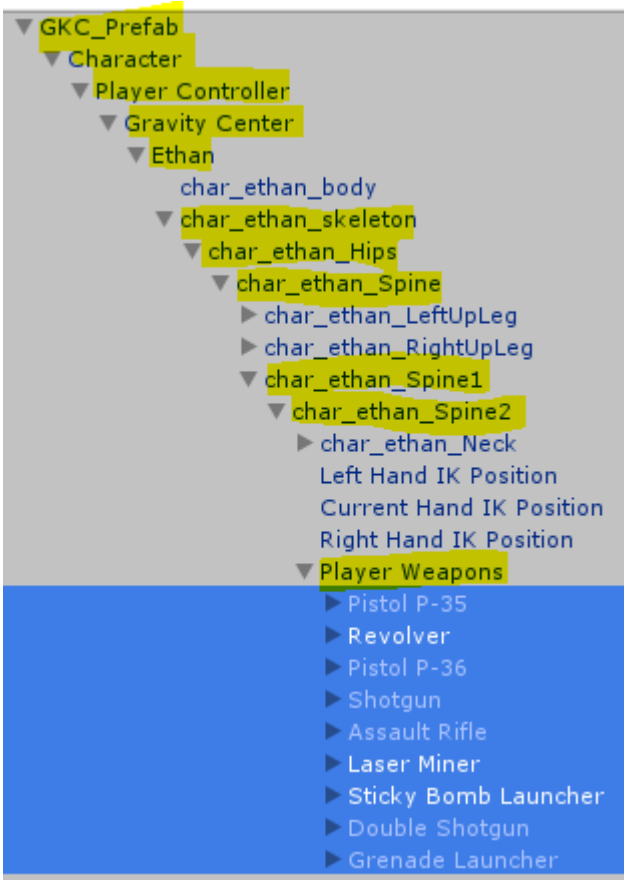
- Press the button Show Weapon List.



- Press the button Disable All Weapons or Clear Weapon List.



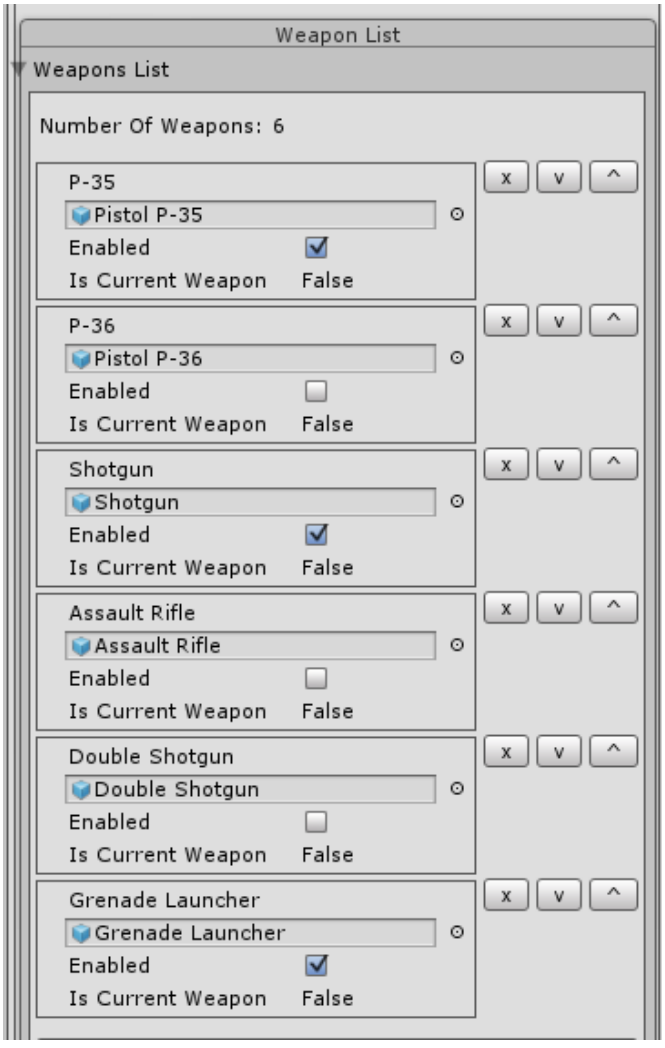
- You can also remove the weapons gameObjects from the player, located inside his body. The image below shows where are located in the hierarchy. Remove all the objets inside the gameObject Player Weapons.



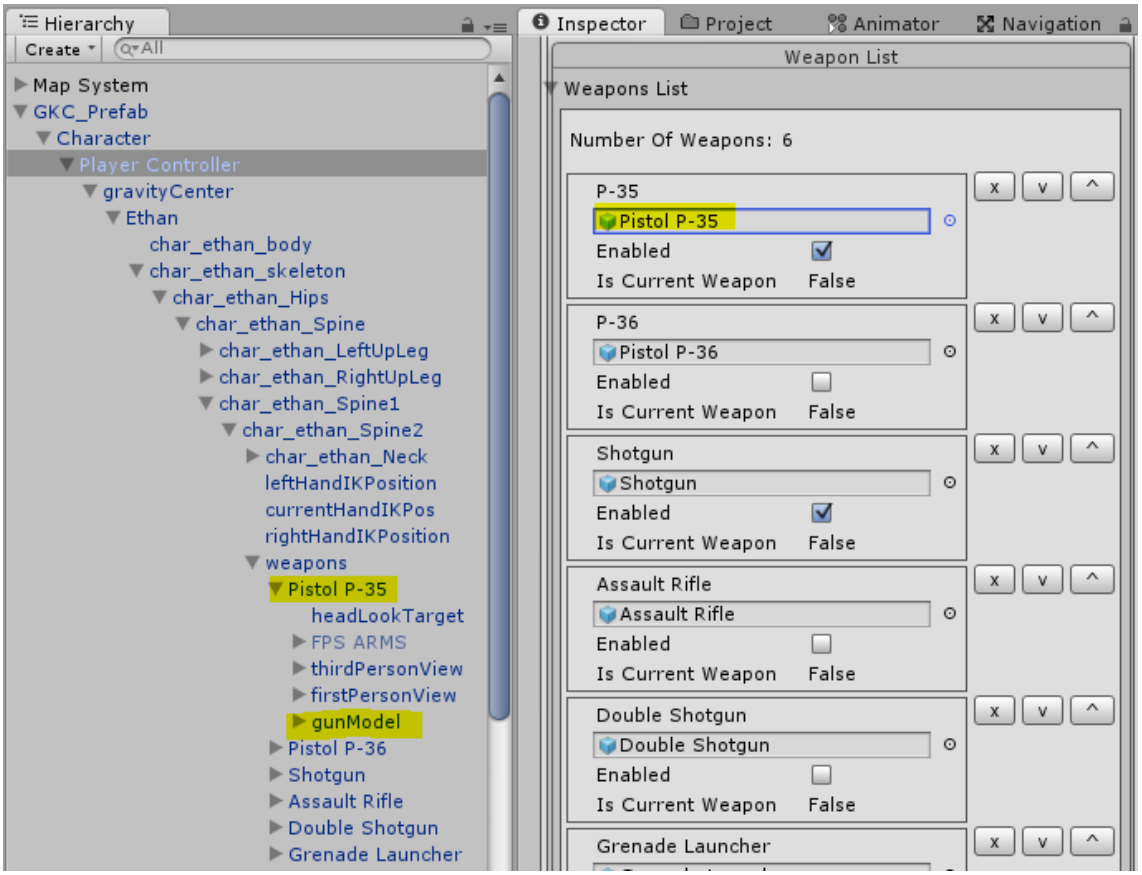
IF YOU NEED TO USE THE WEAPON, follow these steps:

- Press the button Show Weapon List in the Player Weapons Manager.

- You can now press the Enabled (or disabled) option in every weapon to make it usable in game (else the weapon model in the player is disabled, but you can use it if you pick the weapon in the level, but you need first to complete these steps to avoid problems. You can remove every weapon in the list, as explained below, so even if the player finds that weapon to pick in the level, he will be unable to use it).

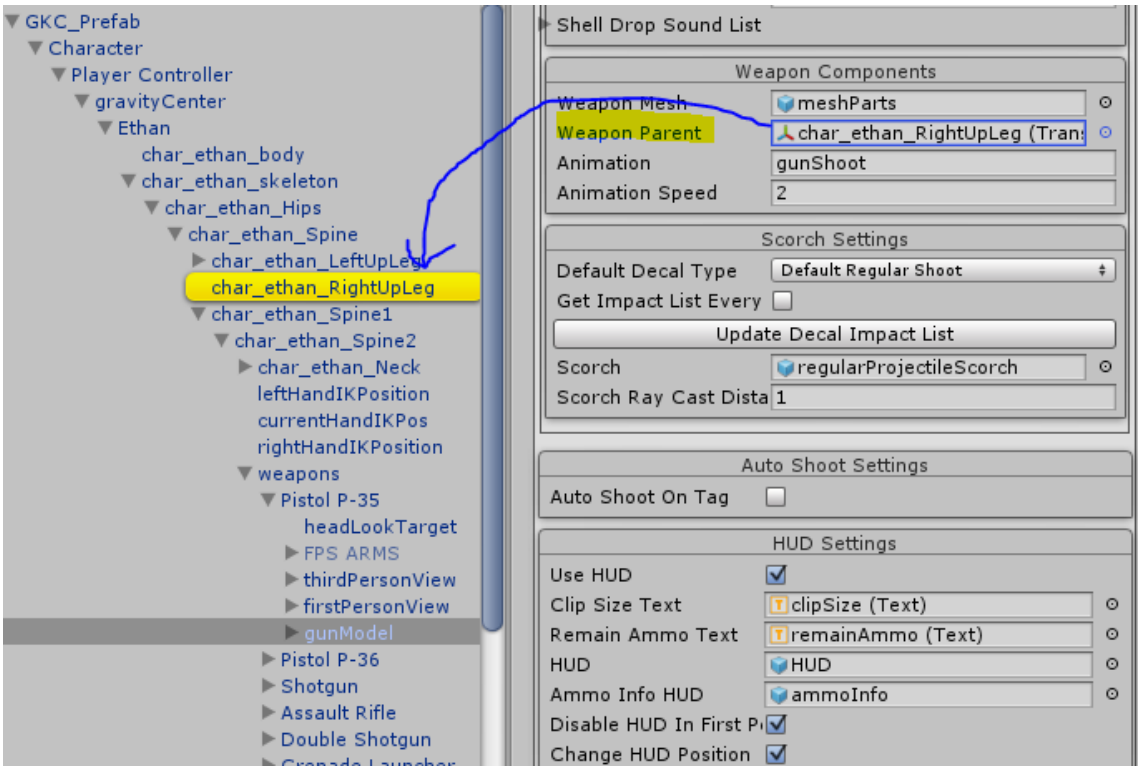


- Finally, you can assign in every weapon where will be attached inside the player's body (the back, the legs, the waist and by default is the back), so if the weapon is in the leg, it is place inside of it. To this, go to any weapon (by selecting any weapon in the above list).



- Go to gunModel inside the weapon, and open the inspector PlayerWeaponSystem.

- Press the button Show Weapon Settings, go to section Weapon Components and set the bone that you want in the label Weapon Parent (for example in the demo, the pistols are parented in every leg and the rest of weapons in the back of the player).

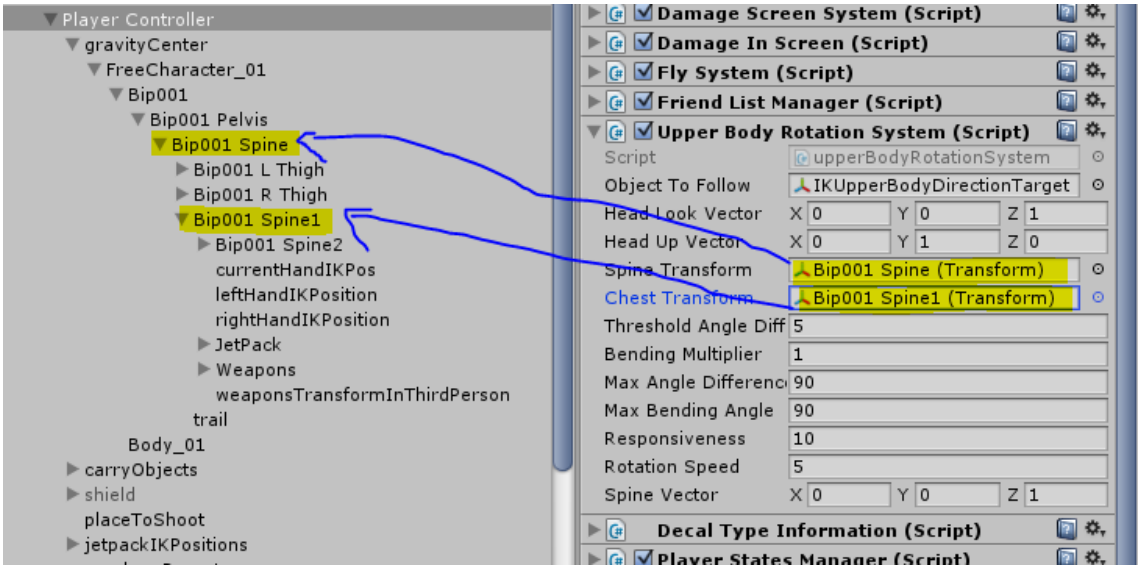

GKC_Prefab
Character
Player Controller
gravityCenter
Ethan
char_ethan_body
char_ethan_skeleton
char_ethan_Hips
char_ethan_Spine
char_ethan_LeftUpLeg
char_ethan_RightUpLeg
char_ethan_Spine1
char_ethan_Spine2
char_ethan_Neck
leftHandIKPosition
currentHandIKPos
rightHandIKPosition
weapons
Pistol P-35
headLookTarget
FPS ARMS
thirdPersonView
firstPersonView
gunModel
Pistol P-36
Shotgun
Assault Rifle
Double Shotgun
Shell Drop Sound List
Weapon Components
Weapon Mesh
meshParts
Weapon Parent
char_ethan_RightUpLeg (Tran
Animation
gunShoot
Animation Speed
2
Scorch Settings
Default Decal Type
Default Regular Shoot
Get Impact List Every
Update Decal Impact List
Scorch
regularProjectileScorch
Scorch Ray Cast Dista
1
Auto Shoot Settings
Auto Shoot On Tag
HUD Settings
Use HUD
Clip Size Text
clipSize (Text)
Remain Ammo Text
remainAmmo (Text)
HUD
HUD
Ammo Info HUD
ammoInfo
Disable HUD In First P
Change HUD Position

ADJUSTING PLAYER'S UPPER BODY FOR WEAPONS

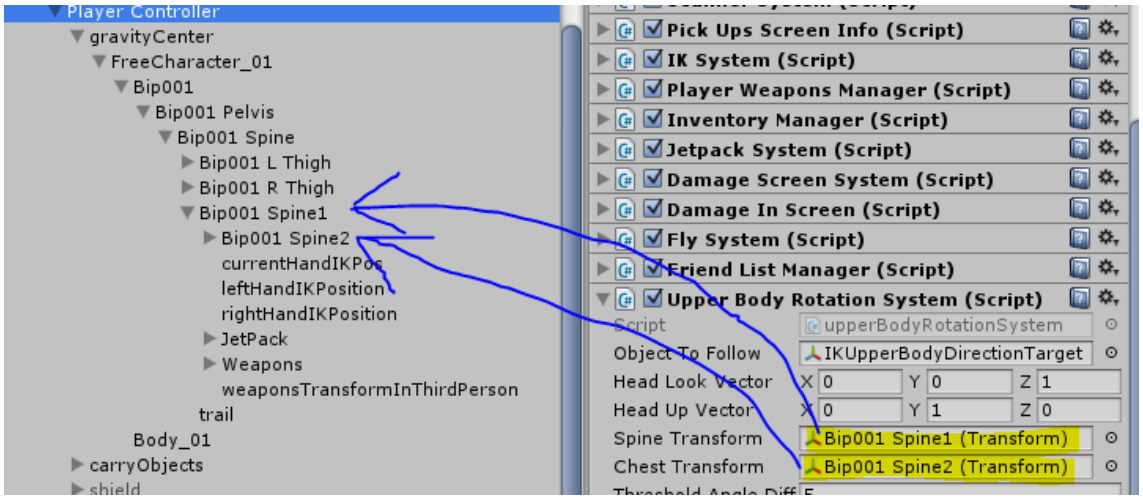
The player's upper body rotation can be configured for aim mode in third person. This is made in the inspector UpperBodryRotationSystem in the Player Controller gamObject, allowing to configure how fast the rotations are done and the clamp values used for the spine rotation. The object to follow is the actual transform followed for the player's torso, which is inside the main camera, so you can move it up or down to adjust the orientation when the player aims.

In the case you configure a new model, and something like this happens:

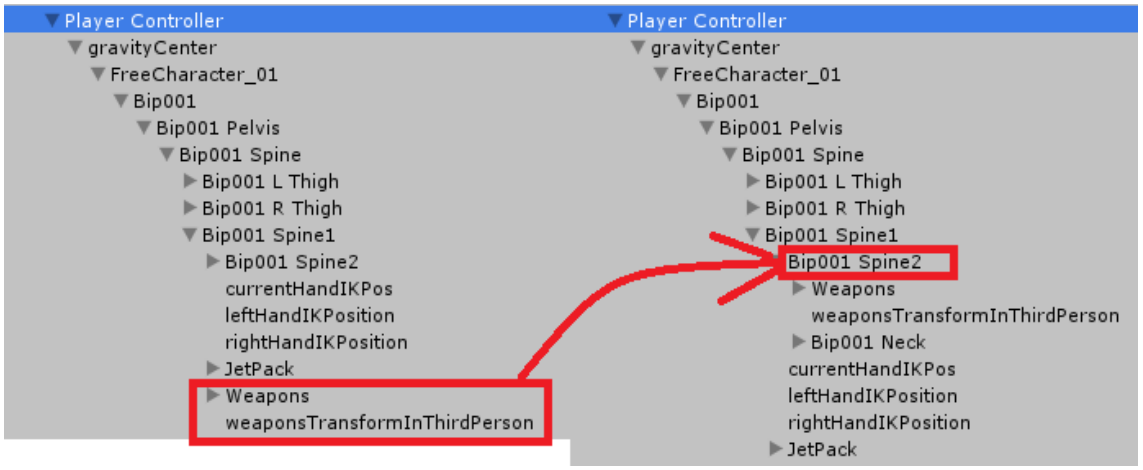
The cause is that the spine bones are not correctly configured in the Upper Body Rotation System, due to the skeleton hierarchy is slightly different in this case. In example above, the system has taken the hips as first bone, this is wrong.



But you only need to set the correct bones. To do this go to the inspector and make sure to place the bones which belong to the spine and the chest, like this:



But if this happens, you need to make another step, which is to move the Weapons and weaponsTransformInThirdPerson gameObjects from its current hierarchy position to inside the Chest transform configured in the inspector, in this case, the Spine2.

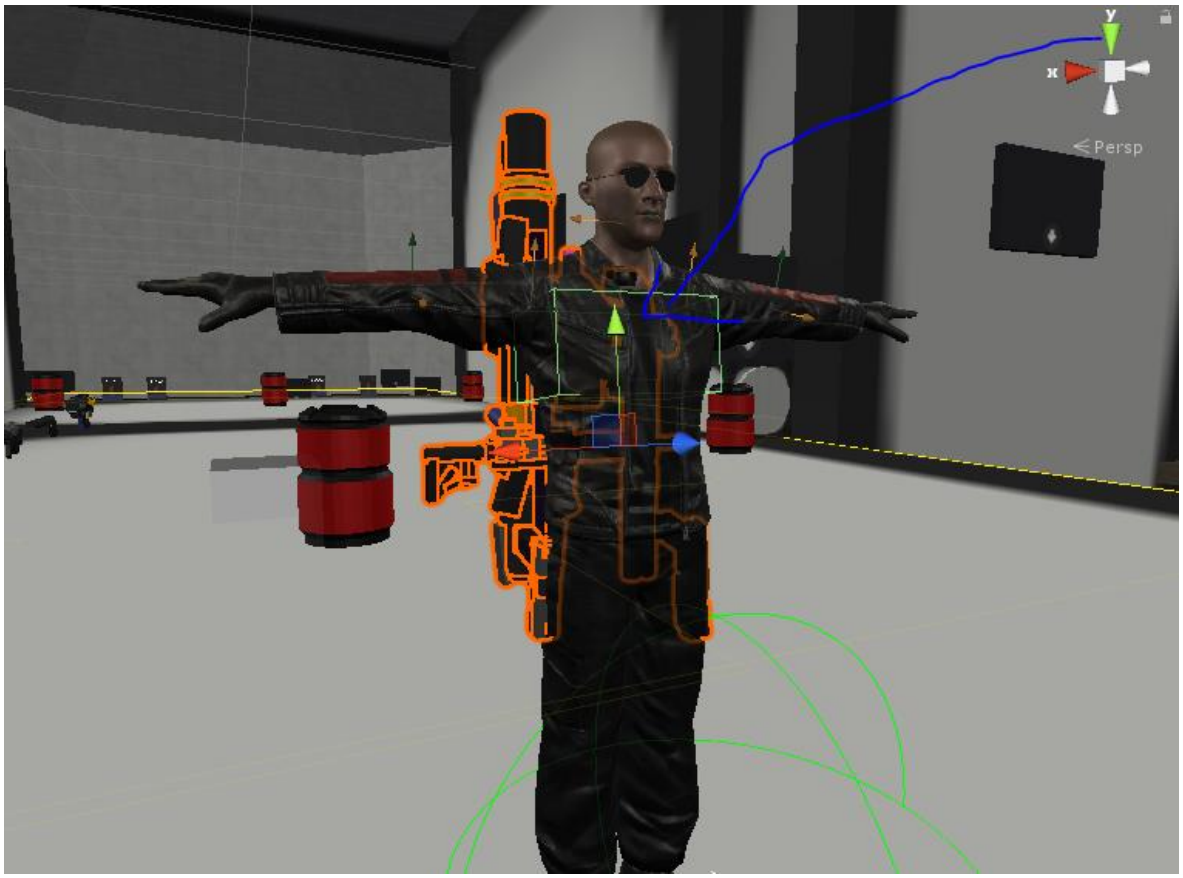


Like this, the player weapon should move correctly, rotating the player's upper body and without any issue.

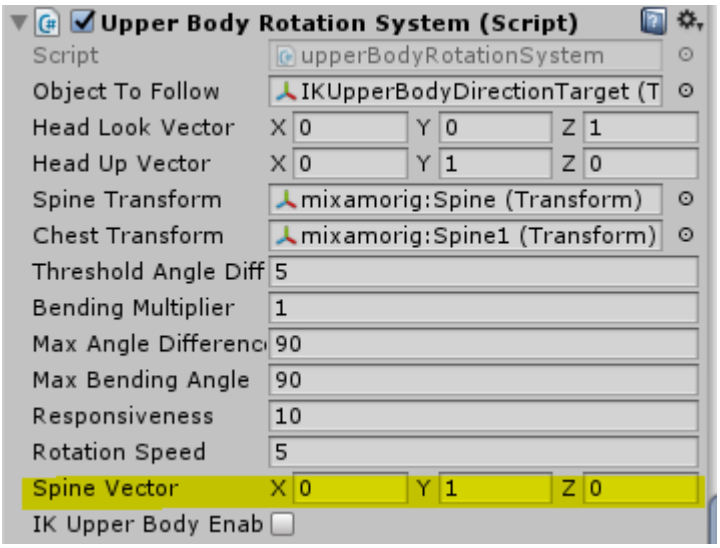
FIXING PLAYER SPINE ROTATION

If the player spine doesn't rotate correctly when you move the mouse up and down in aim mode in third person, the cause is due to the spine orientation in the model.

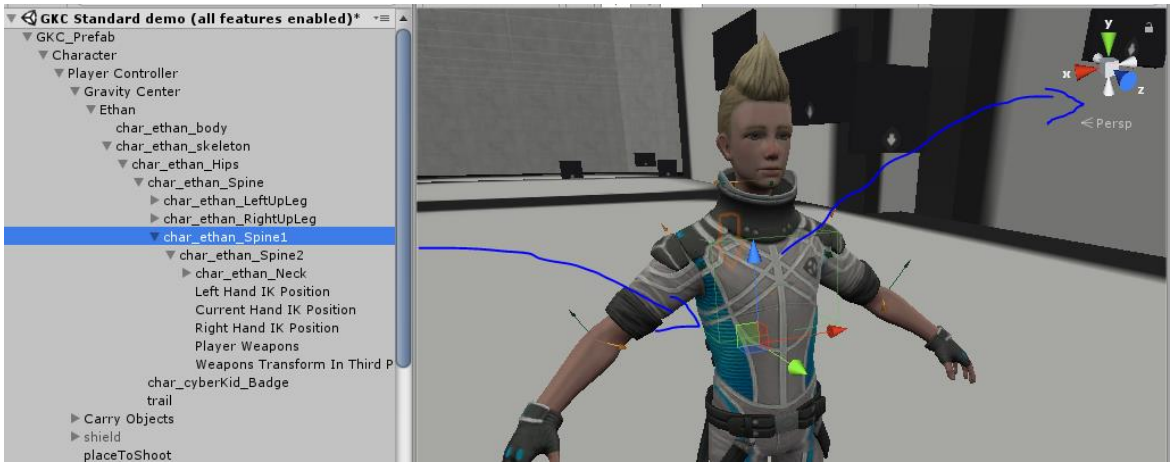
For example, in this model, the local Y axis of this model spine, is the Y axis, which can be different in every model.



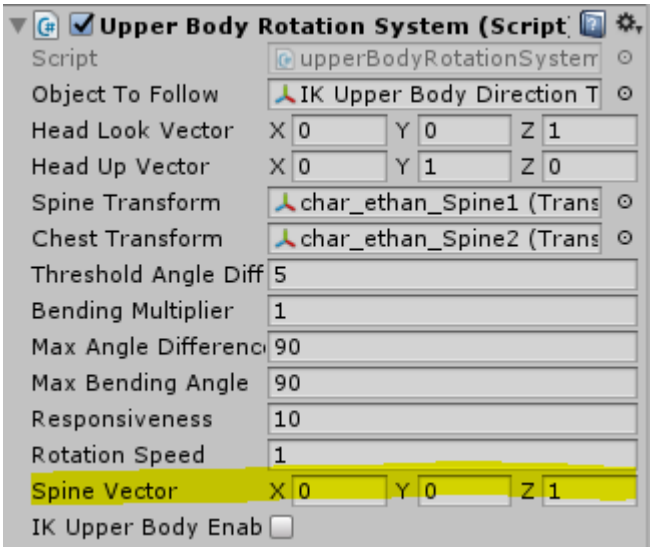
To fix this, go to the Upper Body Rotation System inspector, and configure the correct values in the field Spine Vector. It needs to be the same up transform orientation of the transform assigned in the field Spine Transform (mixamorig: Spine in this example). In this case, the value is (0,1,0) instead of (0,0,1) which is the default value.



In the default model of GKC, its spine orientation is Z for the local Y axis.

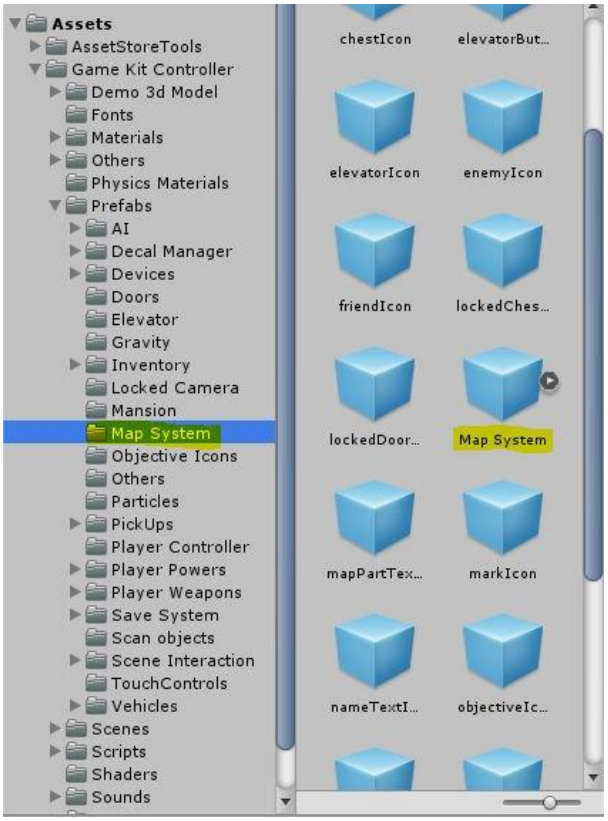


And the value used for this is (0,0,1):

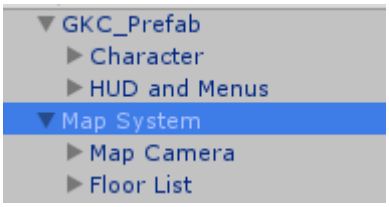


SET THE MAP SYSTEM

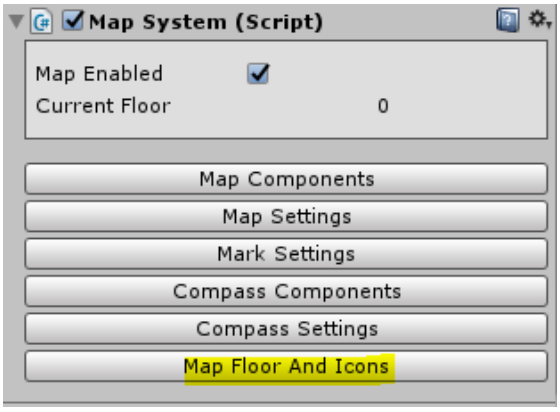
Once that a new player has been created, if you want to use the map system, you need to drag and drop the Map System prefab (with the same name) in the folder Assets/Game Kit Controller/Prefabs/Map System.



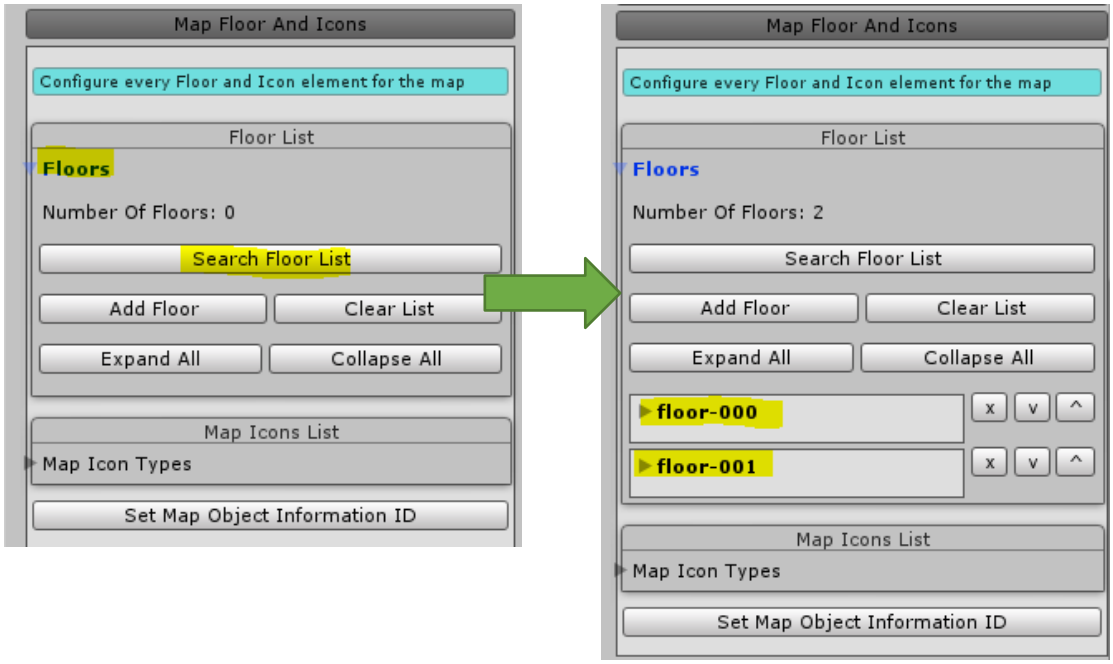
Once the hierarchy contains these two elements:



Then, go to Character gameObject, Map System inspector and press the button Map Floor and Icons.



Go to Floors, and press the button Search Floor List. The floors configured in the Map Creator inspector will be configured automatically.



If you don’t need to use the map system, don’t configure the above steps and it will be disabled.

COMBAT SYSTEM

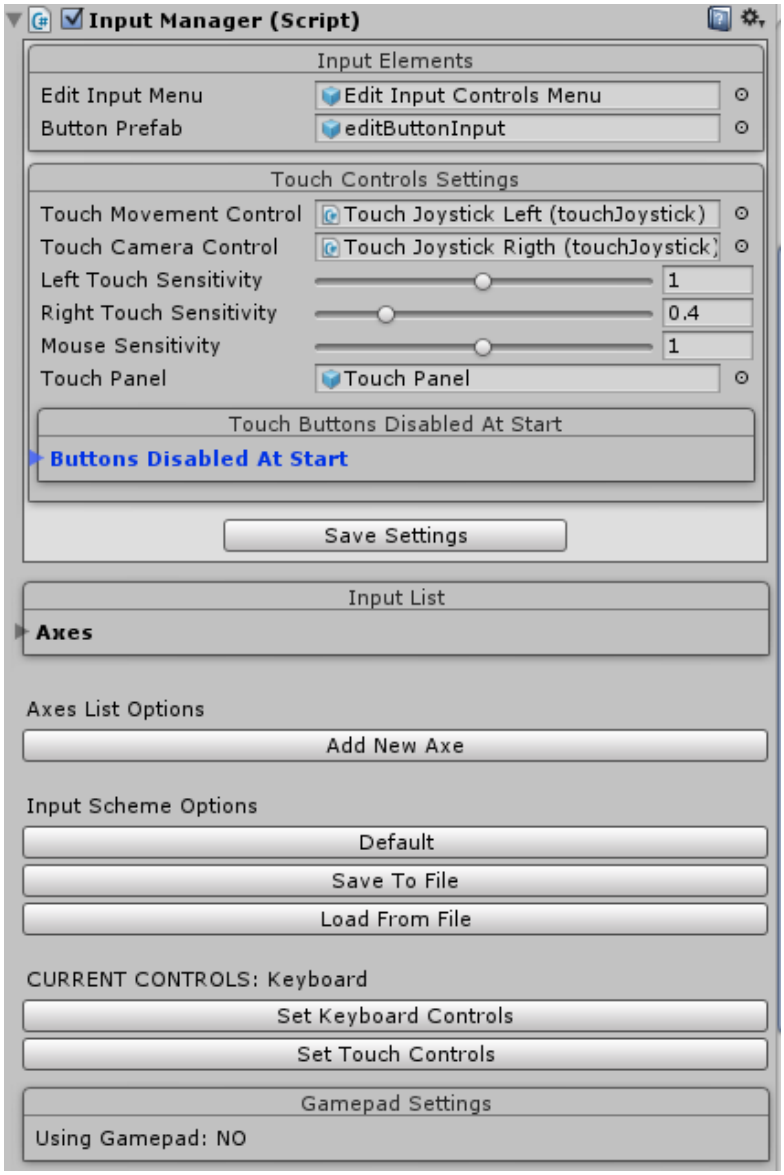
IMPORTANT: THE CLOSE COMBAT ANIMATIONS (KICK AND PUNCHES) ARE NOT INCLUDED IN THIS ASSET, BUT THE SYSTEM THAT USE THEM. YOU CAN FIND THOSE COMBAT ANIMATIONS AND MORE IN THE FREE ASSET TAICHI CHARACTER PACK, WHICH WAS USED TO TEST THIS SYSTEM. TO USE IT, JUST SET THE ANIMATIONS IN THE COMBAT LAYER OF THE ANIMATOR.

EXPLANATION OF EVERY SYSTEM

INPUT MANAGER

MAIN CONTROLS

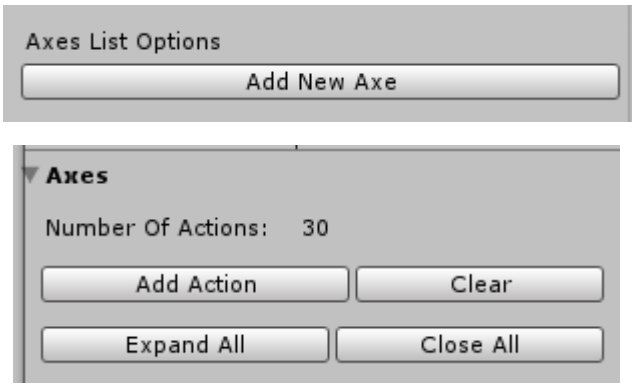
This systems allows to define a list of input keys, save this configuration, load it, and set the default controls, defined inside the script. It uses the default configuration of axes in Unity, so there is no need to configure these original axes or add new ones (only for gamepad and only if the Project Settings are not imported).



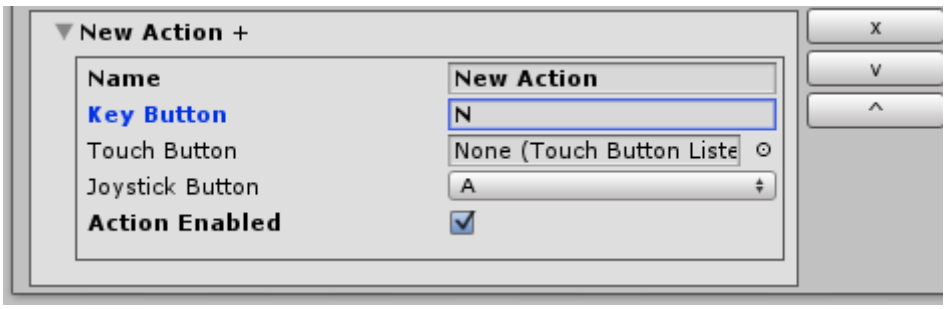
Also, it has the main key detection system, to check if an action key defined in the axes is pressed, held or released, using the get button functions placed in the other scripts. At the same time, it checks the touch controls, asking if the action key has been pressed, held and released. It does the same with gamepads.

Add an action to a new script

- Add a new Axe in the Axes list, by pressing the button Add New Axe or Add Action.



- Set the name that you want for that action and a keyboard button.



- Declare a inputManager variable type (if the script doesn't contain it yet), get the main inputManager in the scene and add the above functions to the script, setting the type of press button and the action that you want to check. In this example you can see it better:

```
using UnityEngine;
using System.Collections;
inputManager input;

public class yourScript : MonoBehaviour {

    void Start () {
        input = FindObjectType<inputManager> ();
    }

    void Update(){
        if (input.checkInputButton ("newAction", inputManager.buttonType.getKeyDown)){
            //KeyDown, key pressed, call the need function
        }

        if (input.checkInputButton ("newAction", inputManager.buttonType.getKey)){
            //Key, key held call the need function
        }

        if (input.checkInputButton ("newAction", inputManager.buttonType.getKeyUp)){
            //KeyUp, key released, call the need function
        }
    }
}
```

Just add this basic code in the script where you want to define an action. The code check for all the three type of input: keyboard, touch devices or gamepad.

if (input.checkInputButton ("newAction", inputManager.buttonType.getKeyDown)){

```
        //KeyDown, key pressed, call the need function
}
if (input.checkInputButton ("newAction", inputManager.buttonType.getKey)){
        //Key, key held call the need function
}
if (input.checkInputButton ("newAction", inputManager.buttonType.getKeyUp)){
        //KeyUp, key released, call the need function
}
```

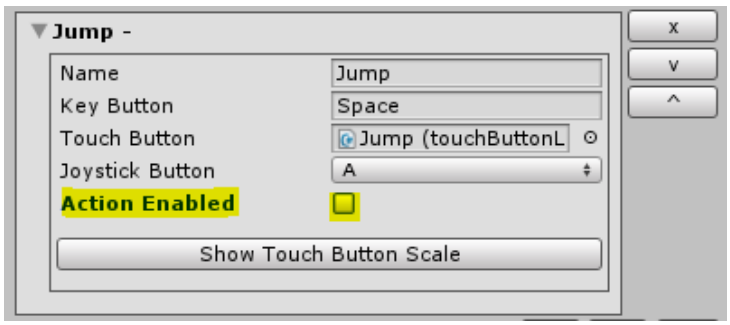
Remove an axe

- Open the axes list and press the X button to remove the action.

If you have this action in the code, there won't be any problem if it is not removed from the script, but it won't trigger any action.

Disable an action

If you want to disable an action but don't want to remove it from the list, unselect the option Action Enabled.

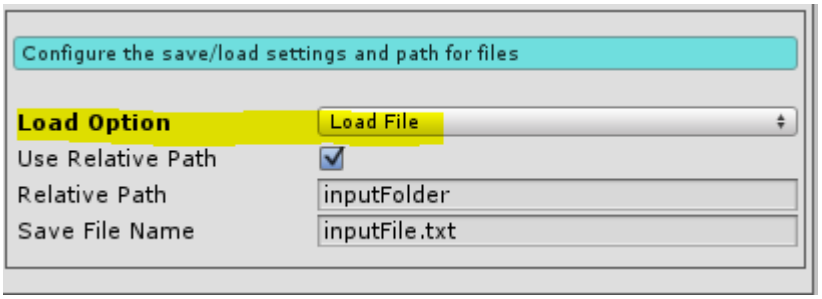


Save/load current axes list configured

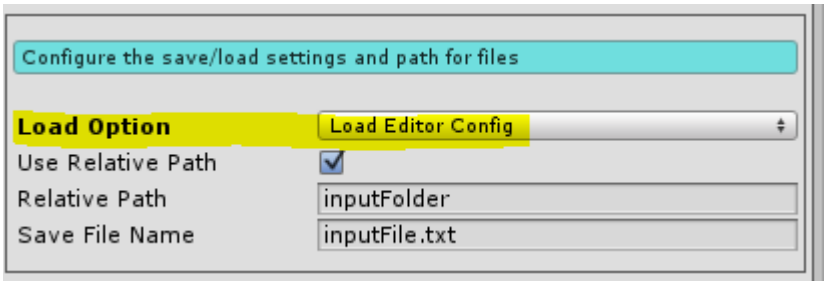
Once you have defined a list of actions, added or remove any action, you will need to save that list. To do this, first you need to configure the save settings (it is already configured by default, but you can configure it):

- Press the button Save Settings.

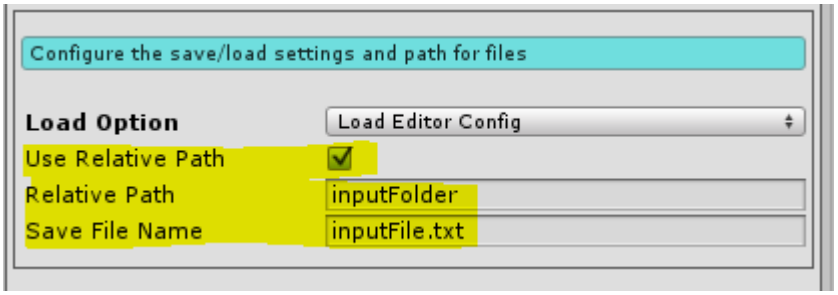
- If you want that the game load the axes from a file when the game starts, set the option Load File in Load Option. The axes list loaded can be different from the list configured in the inspector.



- Else, set the option Load Editor Config, so when the game starts, it will use the current axes configured in the inspector.

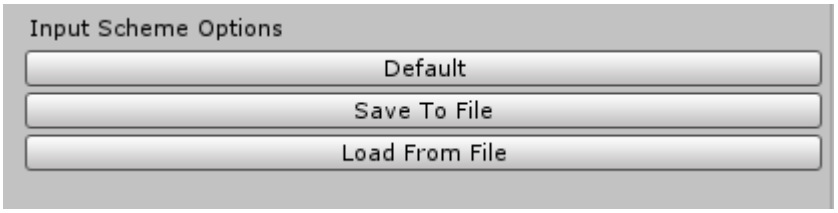


- Also, if you want to use a relative path for the file, just set the option Use Relative Path, set a path in Relative Path and a file name in Save File Name. For this option, the Load Option needs to be Load File, else the axes used will be the configured in the inspector.



Manage axes save file

Once you have selected the path to save the file, you can save the current axes list configured in the inspector, so it will be loaded when the game starts, using as the default configuration.

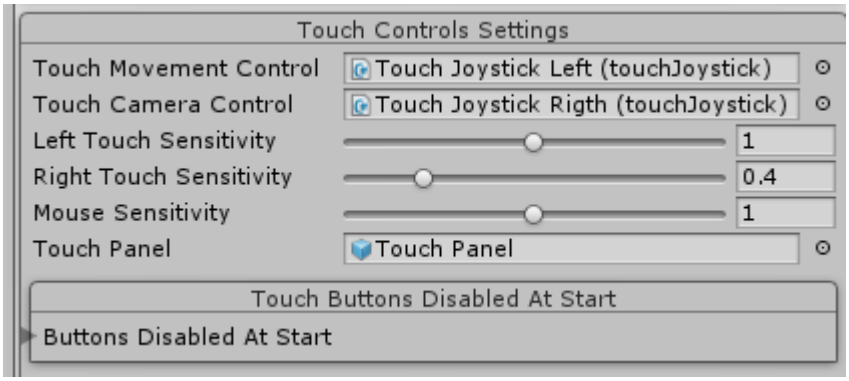


There are different options in the Input Scheme Options:

- **Default**, removes the current axes list and configures it with the controls by default (this is created inside the code, so the default list is always the same, but it can be changed, in the inputManager scipt, go to the function setToDefault and change the list configured). This function also set the touch buttons.
- **Save To File**, save the current axes list in a file with name and path configured previously in the Save Settings.
- **Load From File**, it removes the current axes list and configures it with the controls created in the last save file created.

TOUCH CONTROLS

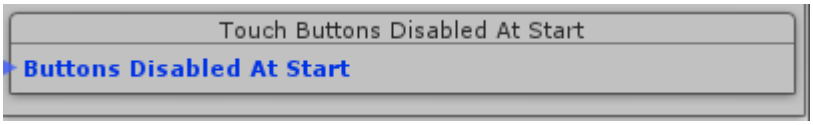
In the Touch Controls Settings, you can configure both touch joysticks for mobile controls, the sensitivity of them and the mouse. Also, here is configured the Touch Panel, which contains every touch button used in touch devices.



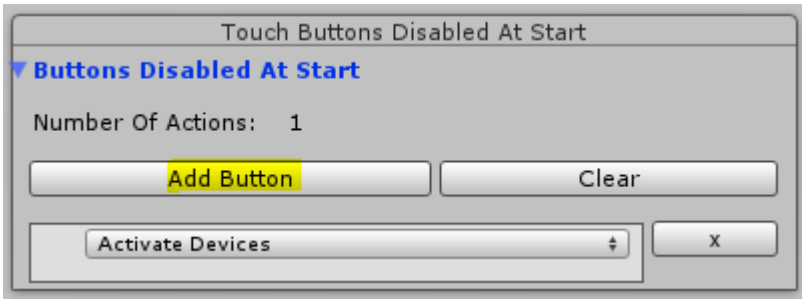
Disable touch buttons at start

If you want to disable any touch button at the start of the game, for example for buttons which are only enabled in certain moments (like the button to open a door or use a device), follow these steps:

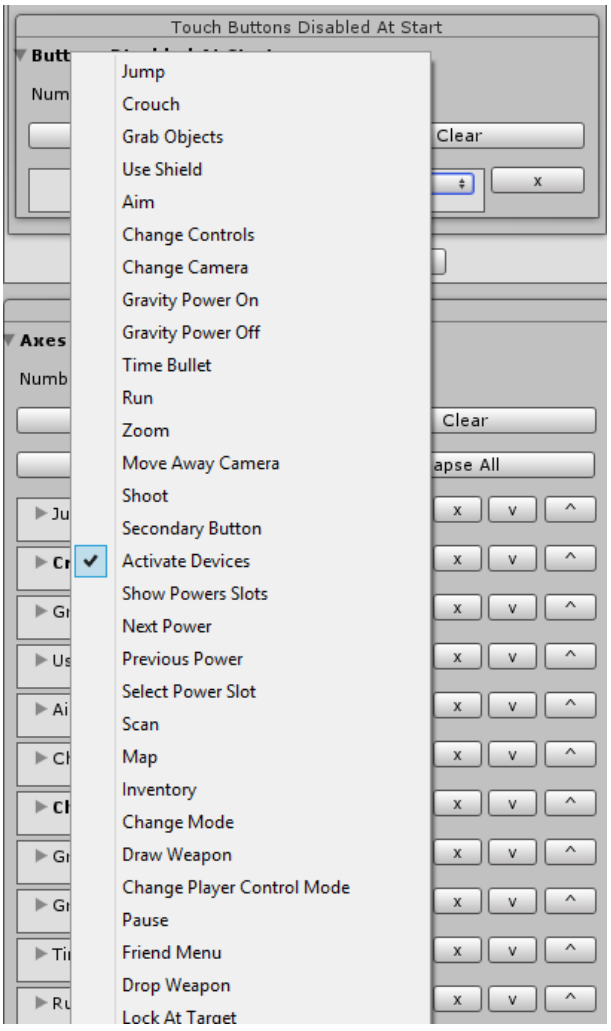
- Open the list of Touch Buttons Disabled At Start.



- Press the button Add Button.



- Select the action whose touch button you want to disable at start (the list of actions is dynamic according to the axes currently configured).



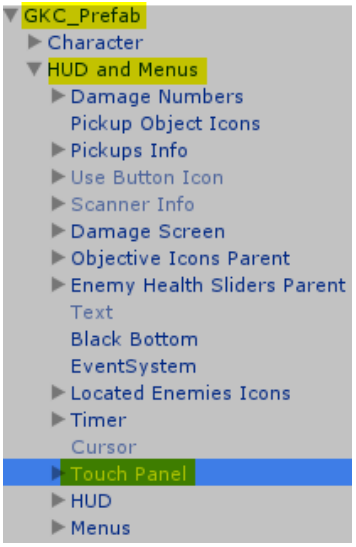
Delete disabled touch buttons

- Press the X button in any element of the list.

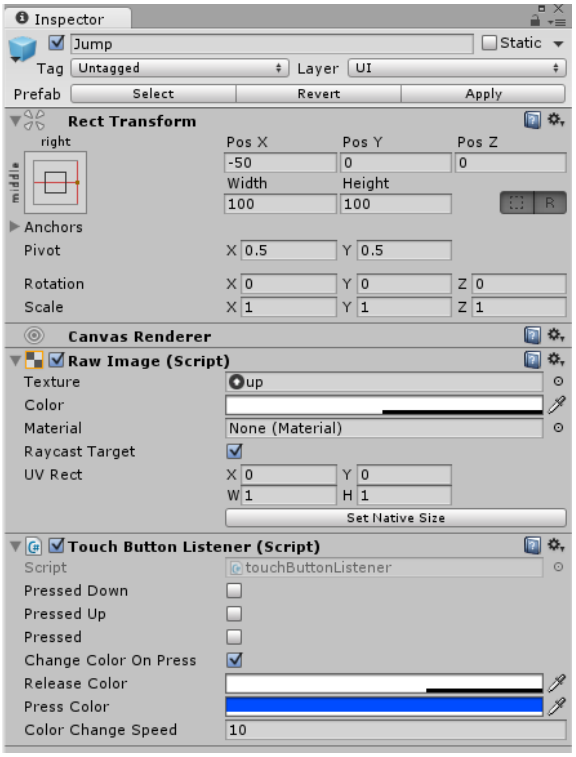
Assign a touch button to an action

Once that you have created an action in the axes list, you can assign a touch button to it:

- Go to the Touch Panel gameObject, inside the Canvas component of the player.

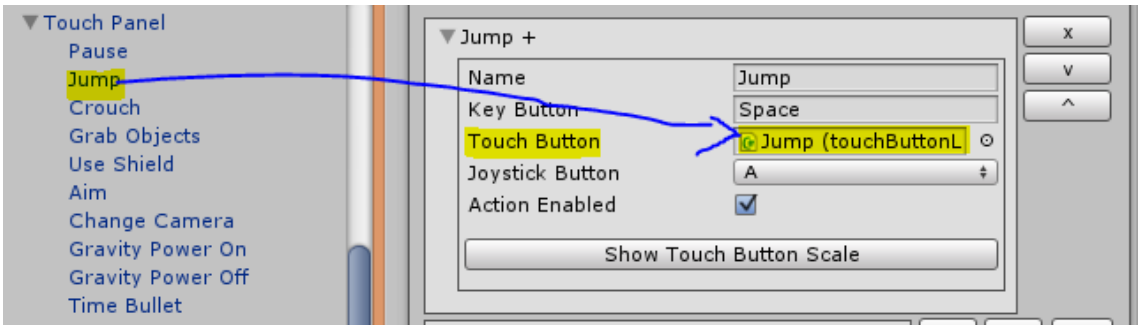


- Copy and paste any touch button inside of the Touch Panel. Also, you can add a new empty object, add a Raw Image component, configure its texture and add a Touch Button Listener. Here how the button inspector looks:



In the Touch Button Listener, it can be configured if the button changes its color on press and release.

- Also, you have to configure the name of that button with the same name that the action which uses it. For example, if the action is called “Jump”, the touch button needs to be called “Jump” too (this is used in the code to identify better every touch button). Finally, assign that object in the Touch Button element of the action configured.

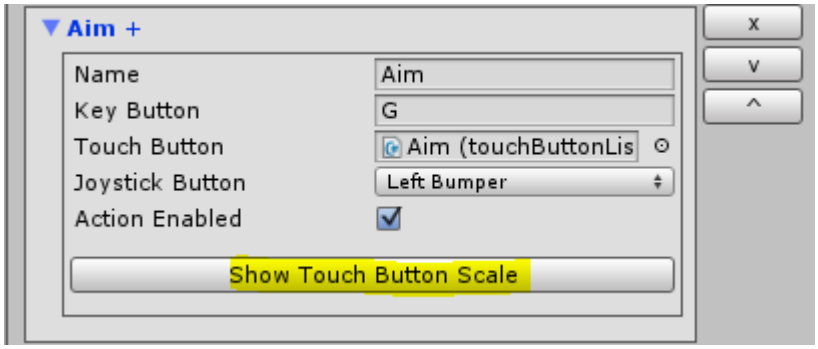


- When a touch button is added, the option Show Touch Button Scale is shown.

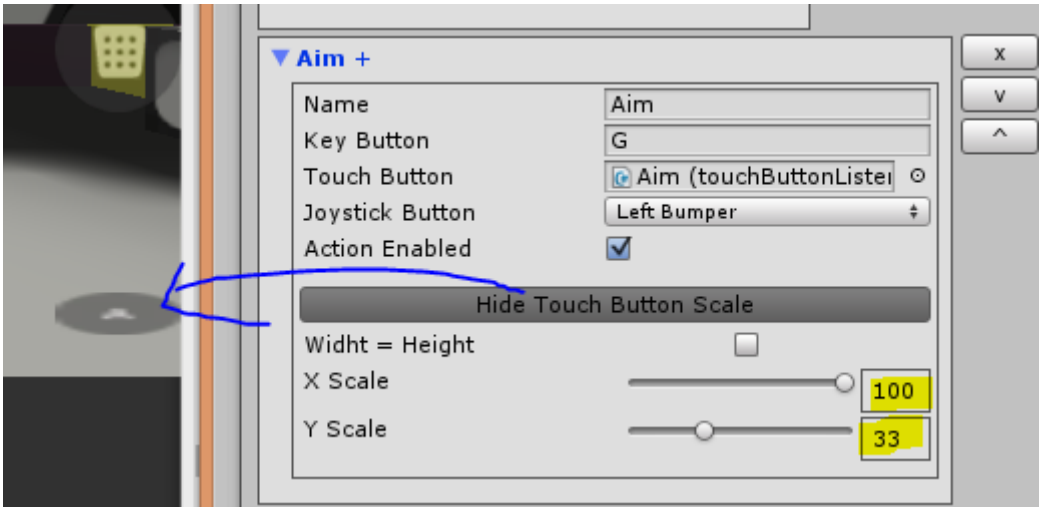
Scale touch button in the inspector

You can change the scale of a touch button from the inspector. To do this:

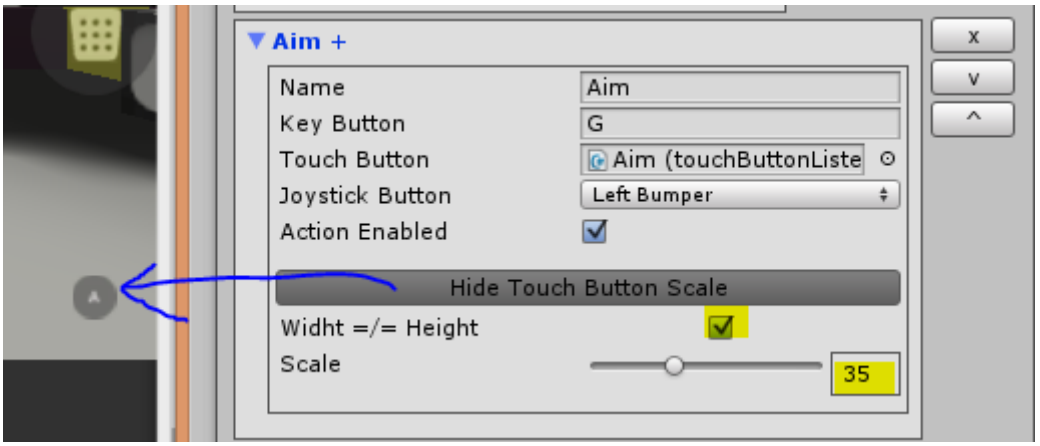
- Press the button Show Touch Button Scale.



- Select the X and Y scale in the range.



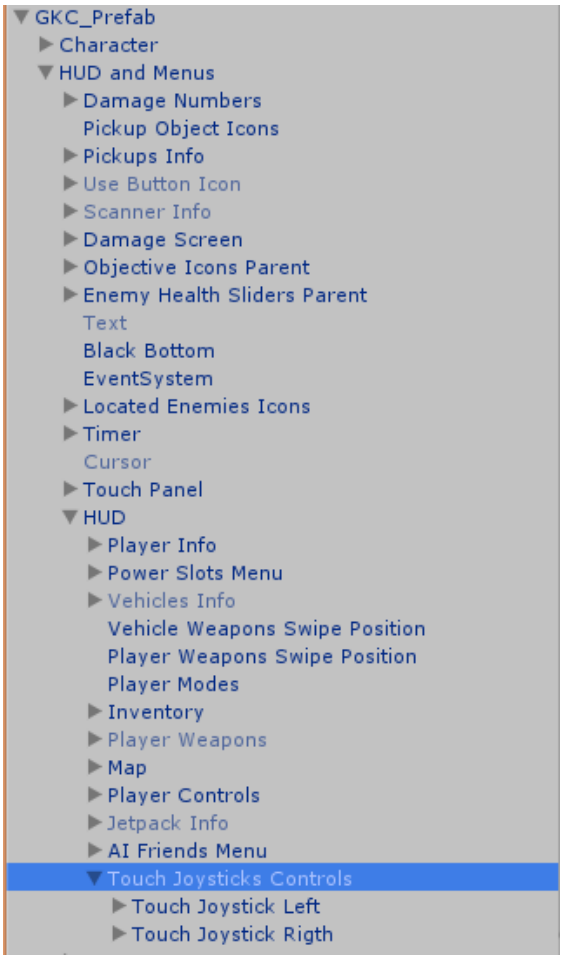
- You can also active the option Width = Height, to set the same value for both sizes.



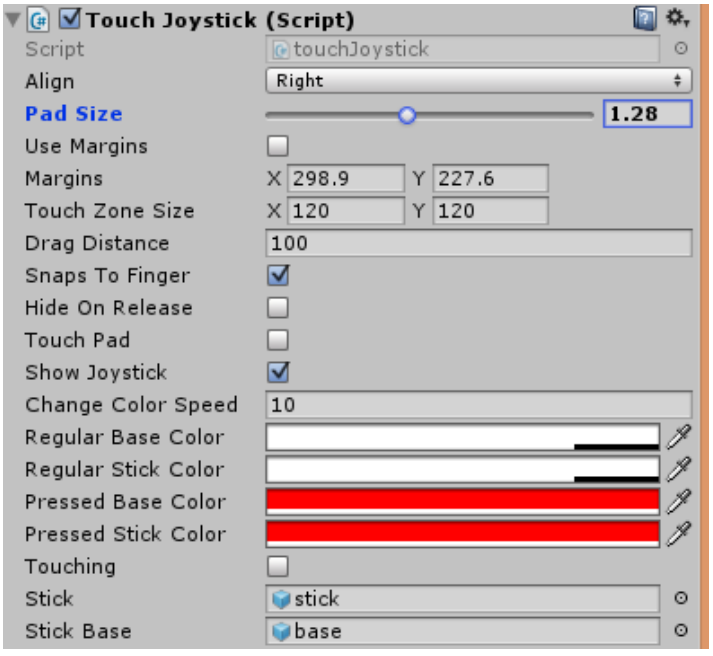
Touch joysticks

The mobile joysticks can be configured with different scales and margins, to adjust better to the screen size according to the touch buttons in the screen. For this, follow these steps:

- Go to Touch Joysticks Controls gameObject inside the HUD gameOject inside the Canvas.

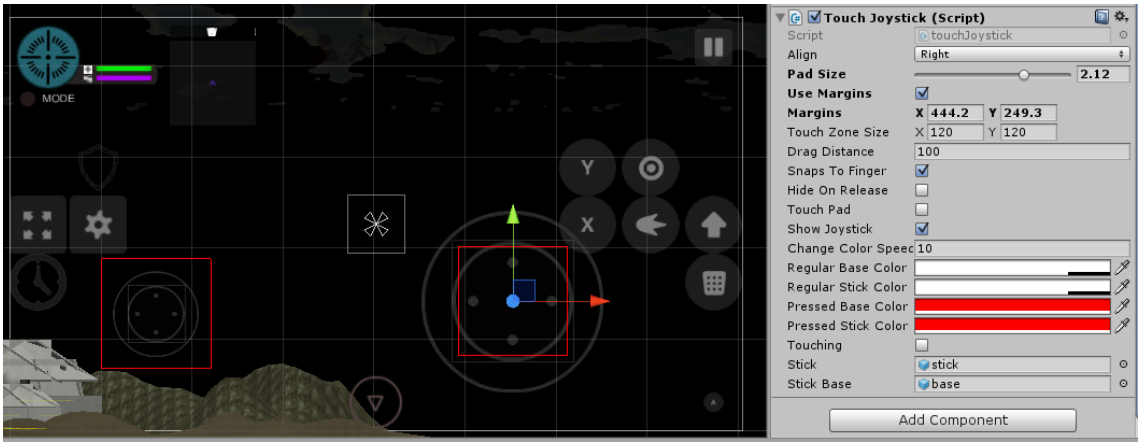


- Select any joystick (right or left) and configure it:



- You can configure the size of the pad, if it uses margins to be placed more to the left or the right, up or down, the touch zone, to check if the player touches the joystick or not, the drag distance, if it snaps to finger, if it hides on release, if it is shown while pressing, if the joytick behaves like a touchpad and if it changes of color while pressed.

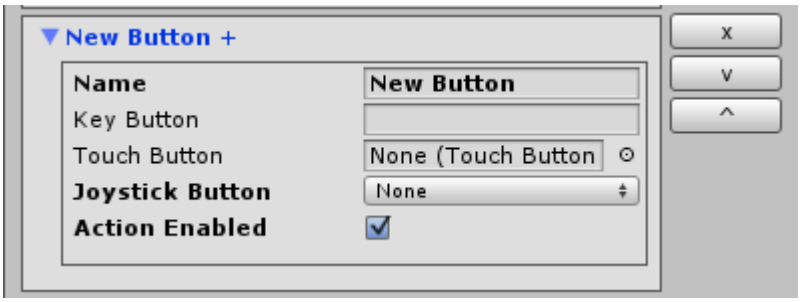
To have a better look at the joystick while it is being configured, use the Scene instead the Game window for it.



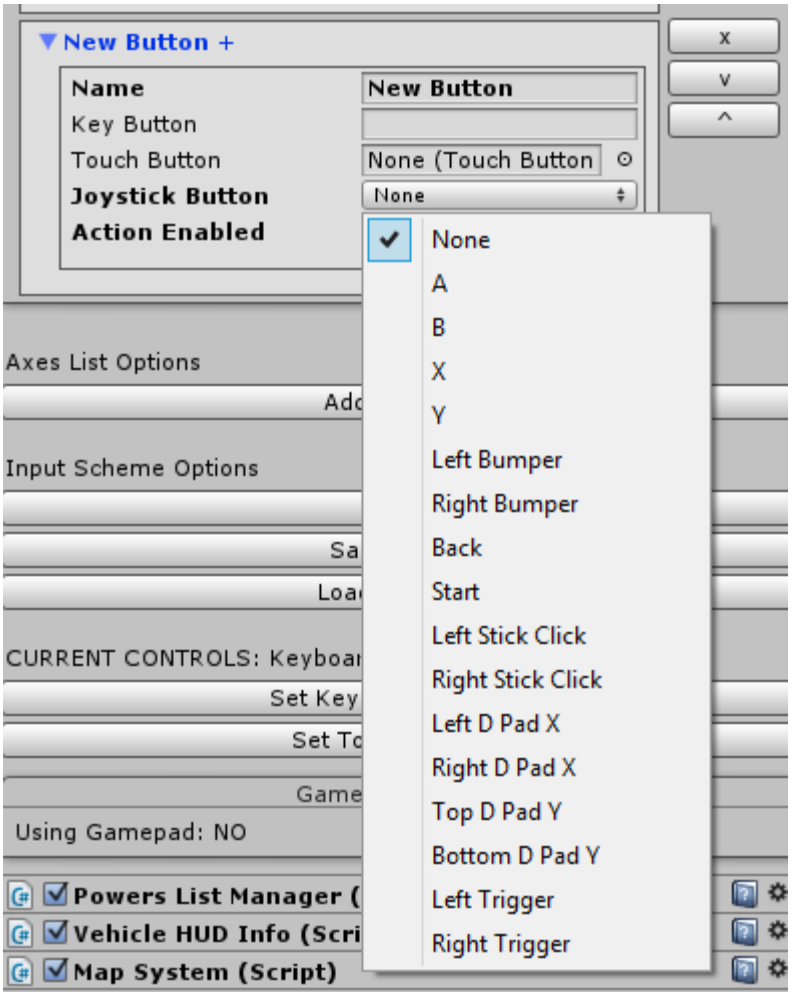
GAMEPAD

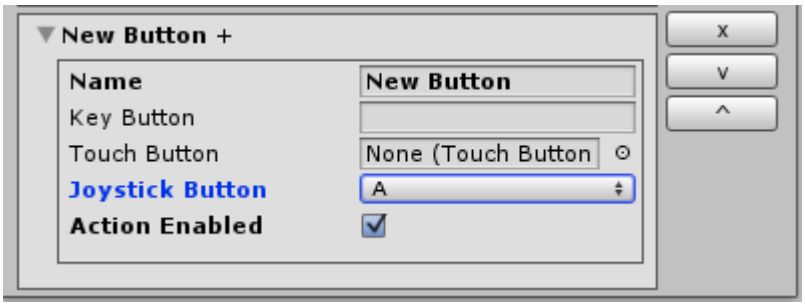
For any action configured in the axis list, you can assign a gamepad button. By default, when you create a new action, it doesn't use a gamepad button. Here you can see in the Joystick Button, which has None configured.

The asset detects automatically if a gamepad is being used or not, allowing to use the keyboard or the gamepad as input.

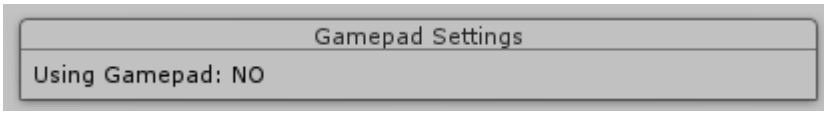


To assign a button, go to the action added and select a gamepad button from the list.





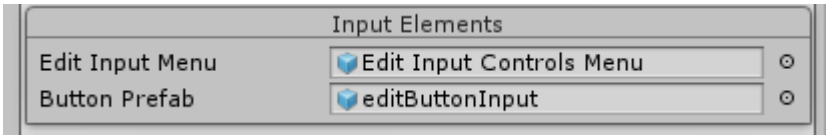
In the inputManager inspector it will be shown if a gamepad is being used or not.



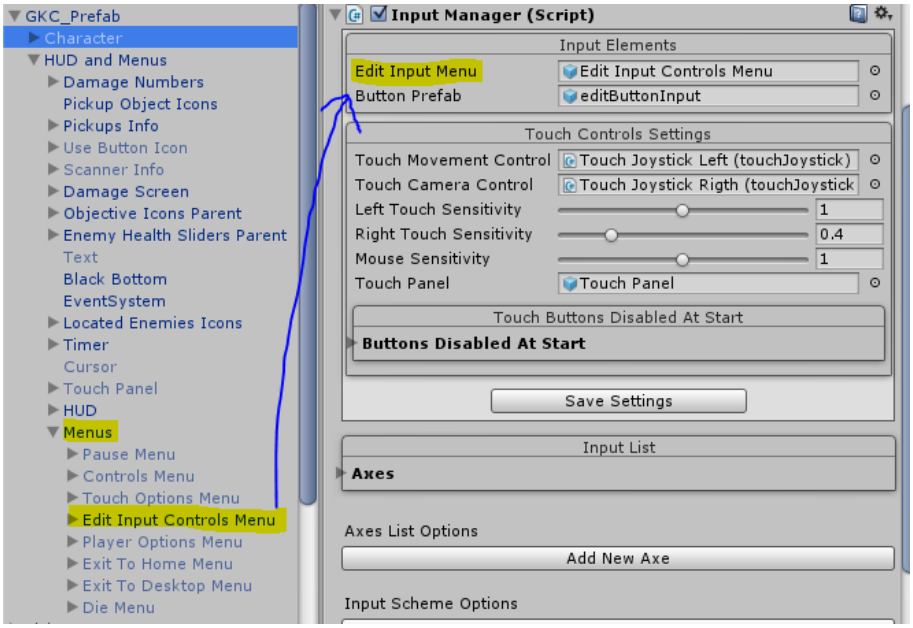
In a future update, the gamepad will have complete use of the menus and any element which uses UI.

INPUT MANAGER IN GAME

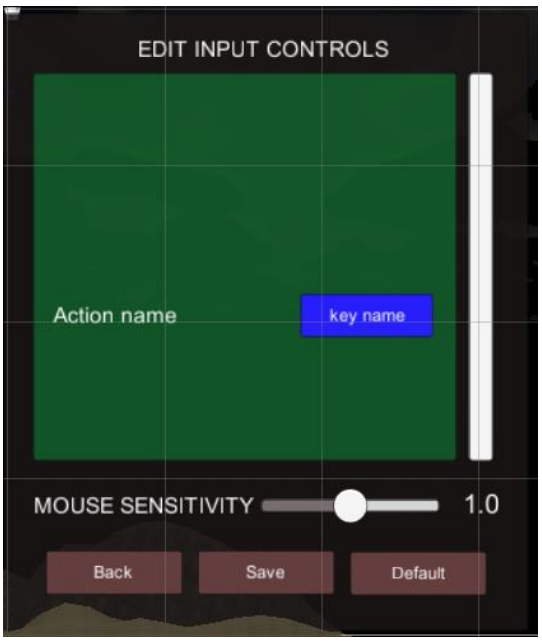
The input manager allows to rebind the actions in game, so you can configure a new keyboard key for every action. The menu for this is configured in the input manager inspector, being the Edit Input Menu element and every action is created using the Button Prefab.



These elements are inside the Menus gameObject inside the Canvas.



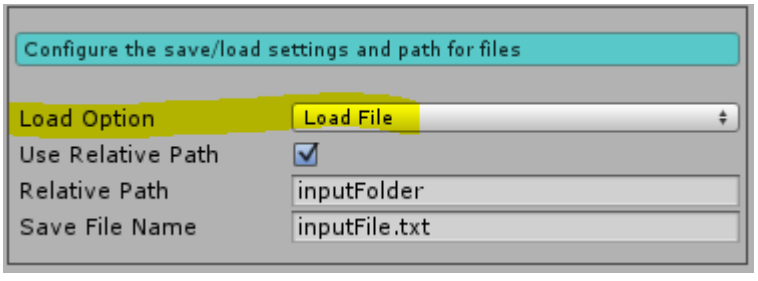
This menu creates dynamically the list of actions according to the axes list used, using a Vertical Layout Group. Here is show how looks in editor mode (left) and in game mode (right).



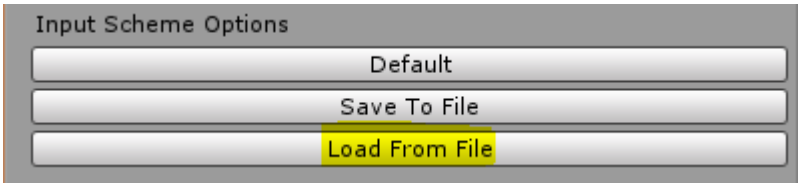
To edit an action in game, press escape, open the Edit Input Controls menu, select an action and press a key. It detects if a key is already used. Finally press Save to save the changed key or back to exit without saving. If you press Default, the default axes list explained above is configured again.

Here you can see the full list of actions in the menu (the mask element used is disabled for this capture):

If any action is changed in this menu and it is saved (using the save button in that menu), the save file for axes is the same, so when the game starts and the option Load File is selected in the Save Settings (image below), the list of action in the file will be loaded, so the changes in the input ingame are saved too.



If you want to update the inspector with the current axes list from the file, press the button Load From file.



CURRENT CONTROLS MODE: KEYBOARD OR TOUCH CONTROLS

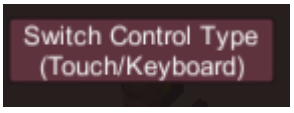
There are two modes of input in the asset when the game starts in editor: Keyboard or touch controls. You can start the game in any of this modes, just press the button Set Keyboard Controls to start the game using the keyboard (The current controls label shows which is selected)

Press the button Set Touch Controls to start the game with the touch buttons and joysticks enabled (these elements are enabled or disable in editor time too).

This is made for debug and testing purposes, so you can test the game in the editor with any type of control, for PC or mobile.

You don't need to press a control or another when you build a game, the asset automatically detects platform and enables the correct type of control.

In the current axes list there is an action to change between these controls mode (C by default) if you are using the keyboard. If you are using the touch controls, pause the game and press the button Switch Controls Type.



PLAYER CONTROLLER

- RAGDOLL
- LOCKED CAMERA MODE

PLAYER CAMERA

- HEADBOB
- SHAKE CAMERA SYSTEM
- LOCKED CAMERA SYSTEM

HEALTH SYSTEM

SCANNER SYSTEM

VEHICLES

- HOVERBOARD WAYPOINTS
- IK DRIVING SYSTEM
- INPUT ACTION MANAGER
- SKID MANAGER
- VEHICLE HUD INFO
    - CAR
    - MOTORBIKE
    - HOVERCRAFT
    - HOVERBOARD
    - AIRCRAFT
    - SPHERE
    - USABLE TURRET
- GRAVITY SYSTEM
- WEAPONS

AI

INTERACTION ELEMENTS

- EXPLOSIVE BARRELS
- CRATES
- DOORS
    - TRIGGER
    - BUTTON
    - LOCKED
    - SHOOT
    - HOLOGRAM
- PRESURE PLATE
- HEALTH/DAMAGE TRIGGERS
- MOVING PLATFORMS
- FALLING PLATFORMS
- TELEPORTATION PLATFORMS
- LASERS
- OBJECT ON RAILS
- PLAYER WAYPOINT PATH
- ZIPLINES
- ELEVATORS
- HEAL CHAMBER
- SCAN OBJECTS
- VENDING MACHINES

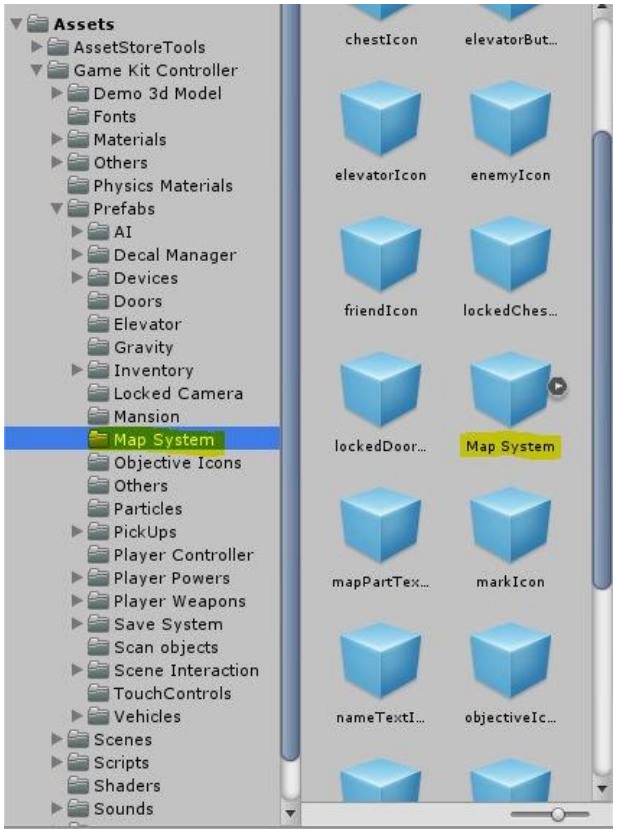
DEVICES

- EXAMINATE OBJECTS
- CODE TERMINAL
    - HACKABLE CODE TERMINAL
- SECURITY CAMERAS
- COMPUTER DEVICES
- MOVE CAMERA TO DEVICE
- MOVE DEVICE TO CAMERA

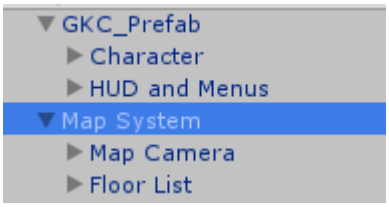
MAP SYSTEM

Create new Map Creator and add map parts

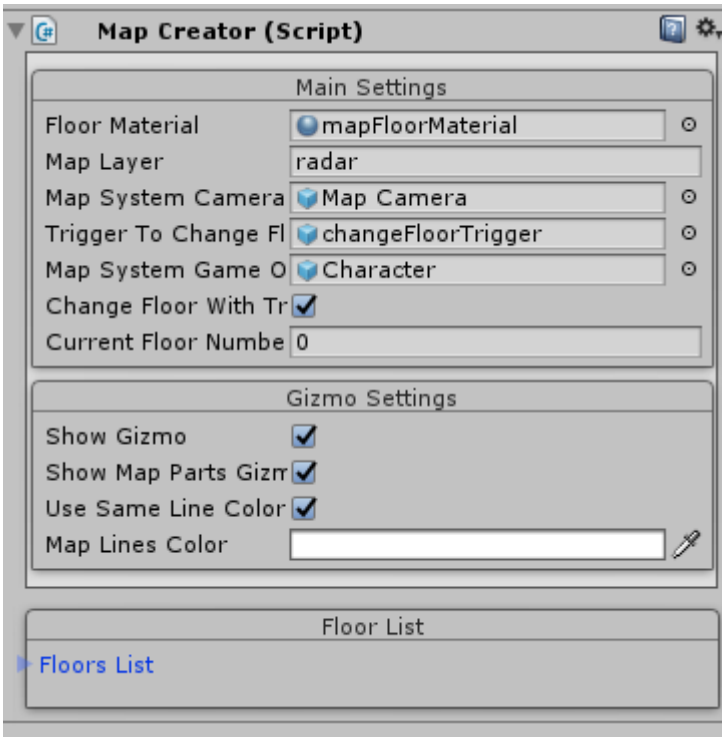
If you want to use the map system, you need to drag and drop the Map System prefab (with the same name) in the folder Assets/Game Kit Controller/Prefabs/Map System.



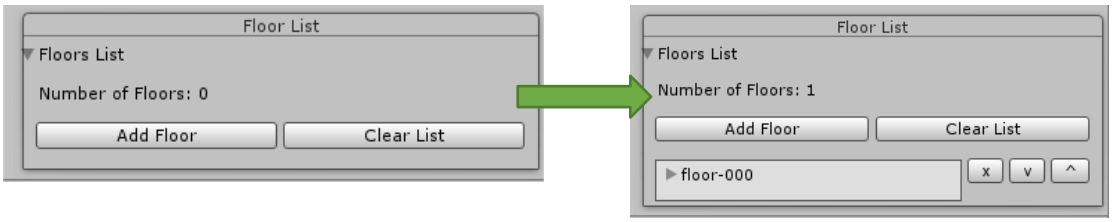
Once the hierarchy contains these two elements:



Inside the hierarchy of the Map System gameObject, go to Floor List and to Map Creator inspector.

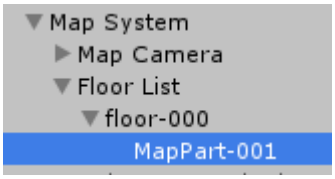


Go to Floors List and press Add Floor button.

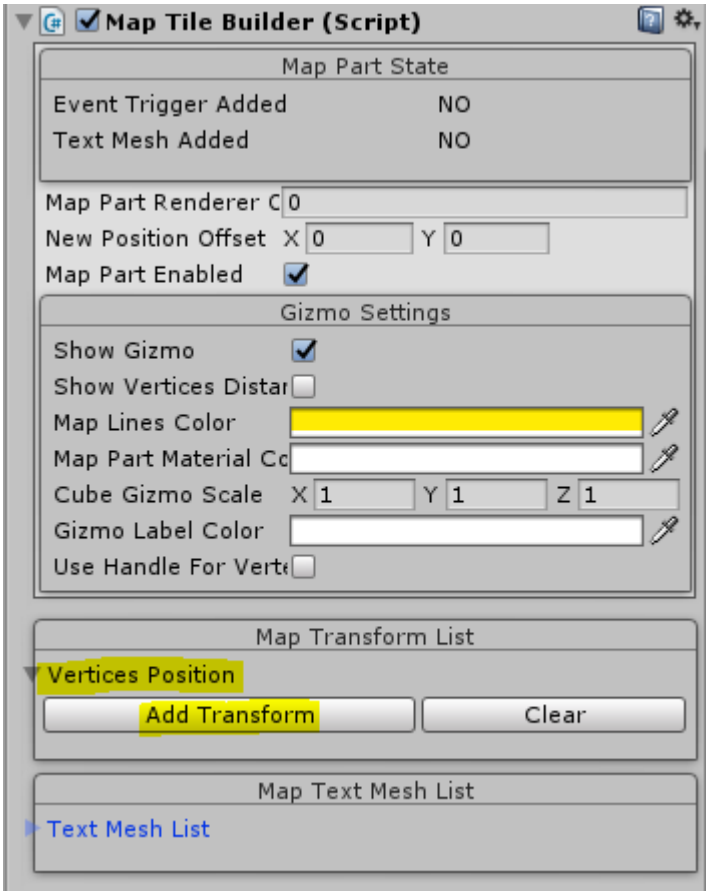


This element will contain all the map parts for the same floor. At the same time, every map part contains a group of empty transforms which are used as vertex to create the map part mesh, like a room, a corridor, a wall, etc…

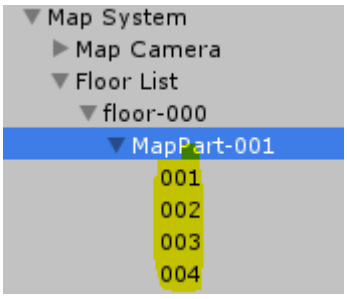
In the hierarchy, you can see that inside the Floor List transform, a new transform has been added as a parent to that floor and inside that floor, the first map part.



Open the first element of the floor list, which already contains the first map part, to the Map Tile Builder inspector, and press the button Add Transform in the Vertex Position list.

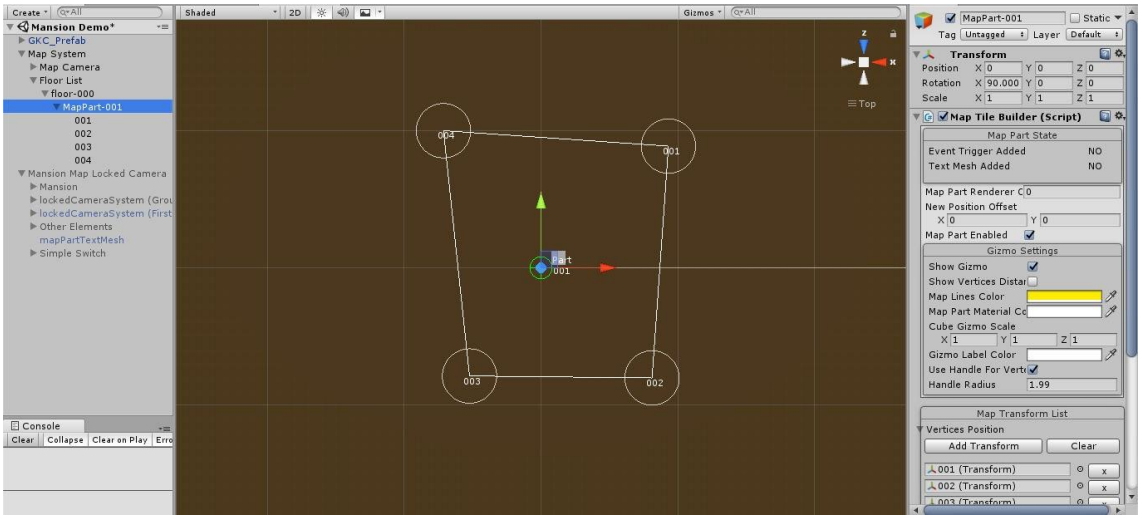


For example, if you press this button 4 times, 4 empty transform will be attached inside the MapPart-001, used as vertex for the mesh that will be created.

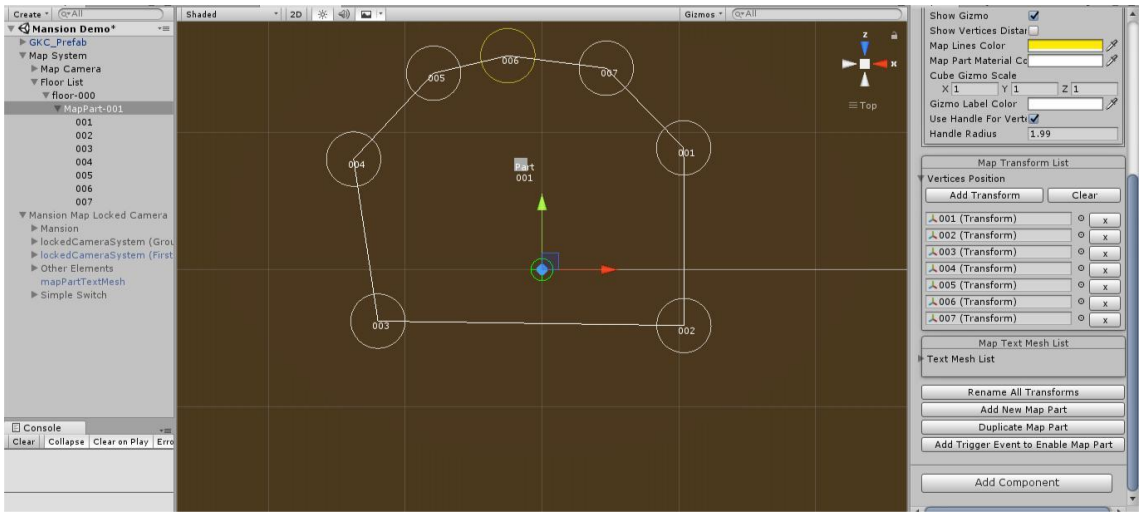


For an easy workflow, in the Map Tile Builder, go to Gizmo Settings and enable the Use Handle For Vertex and configure the radius as you need.

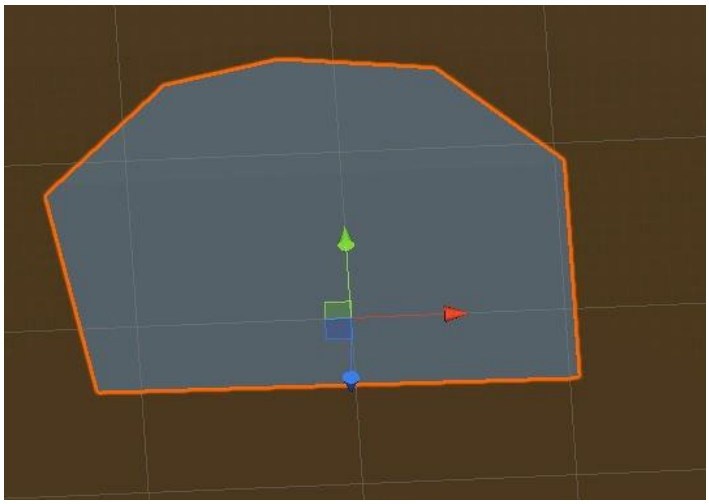
You can also move the transform 001 to 004 to place the vertex to make the shape that you need.



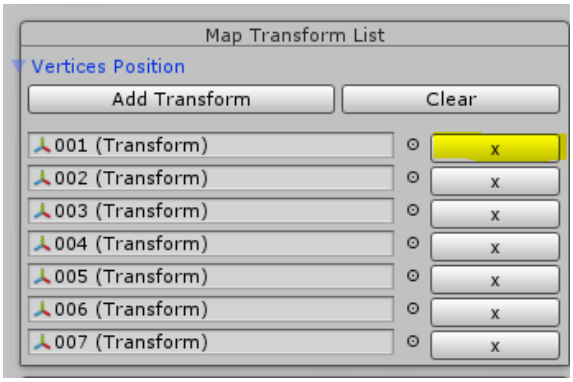
You can add as many vertex as you need, to make rooms or places simple, for example squares or rectangles to circular rooms, hexagonal, etc…, just press the button Add Transform.



When the game starts, the shape is created, looking like this:

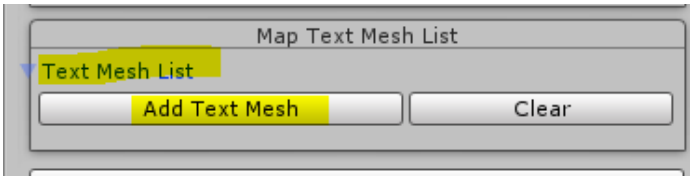


You can remove any of these vertex by pressing the X button and both the object in the scene and in the list are remove.

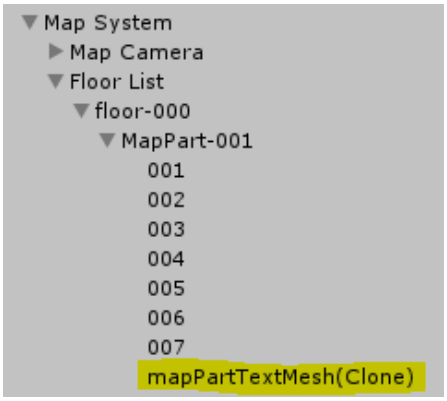


Add name mesh for the floor of the map part

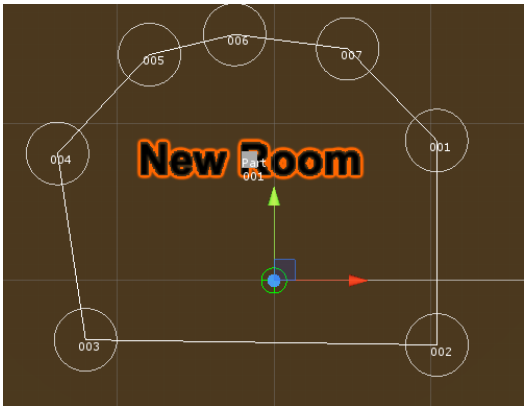
This can be used to place names in any room or to place the floor name shown in the map. For this, go to Text Mesh List in the Map Tile Builder inspector and press Add Text Mesh.



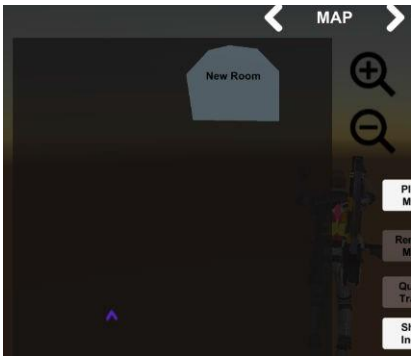
A new text mesh is attached inside the map part transform.



Configure in the inspector the size and content and place where you need.

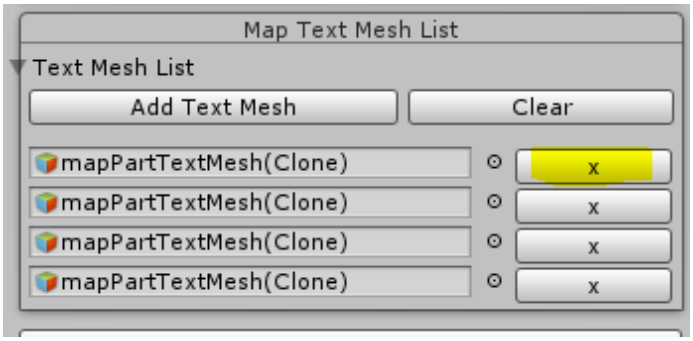


If you press play, the text will be shown in both minimap and map window menu.



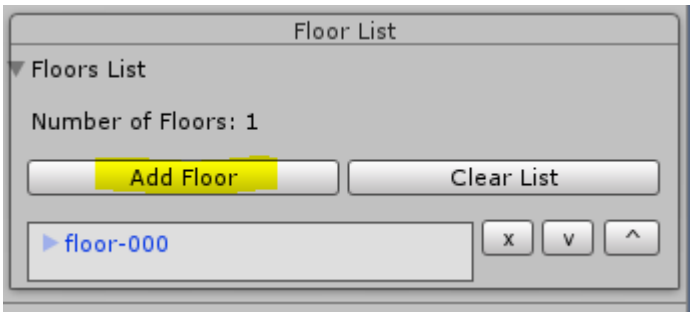
You can add as many text mesh as you need in every map part. You can see in the gizmo, that every text mesh is linked with its map part owner.

To remove any of these mesh, press the X button and both the object in the scene and in the list are remove.

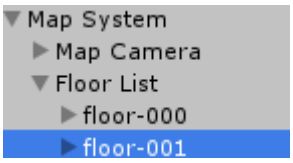


Manage multiple floors

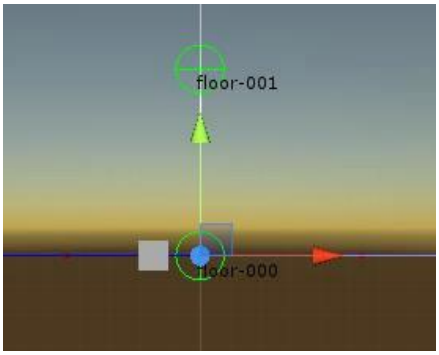
Go back to the Floor List gameObject, to the Map Creator Inspector, and press the button Add Floor again.



Like before, a new floor transform will be added in the hierarchy.

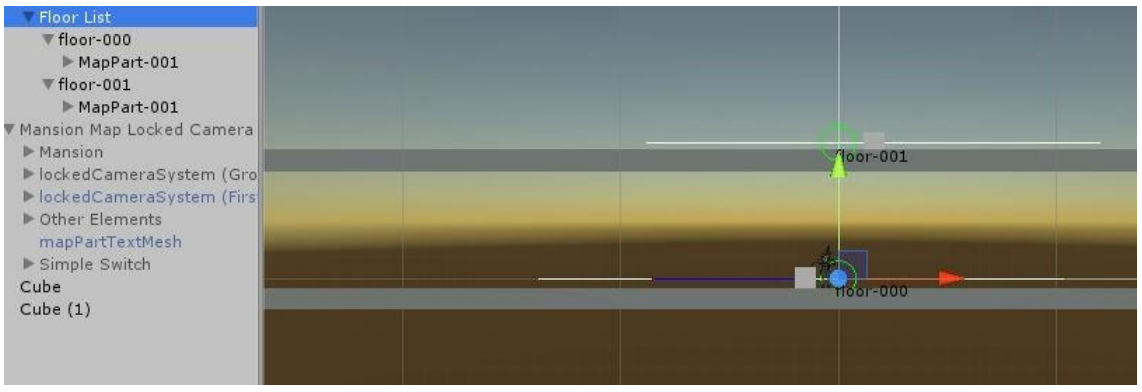


Also, in the scene, a gizmo shows the position of every floor, so move the new floor position according to your level scene.



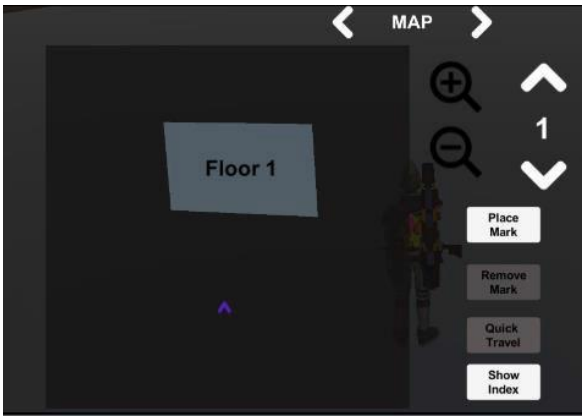
This will be used to know where the player is according to the closest vertical distance to the floor list (used by default). Now you can start to add new map parts to the new floor.

For example in this image, there are two floors using two cubes and two map floors configured, so the closest floor map to the player will be rendered in the map window.



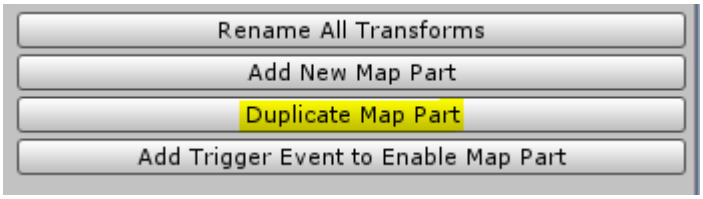
With the new floor added to the list, don't forget to go to the Map System inspector and press Search Floor List.

And this is shown in the map menu by floor:

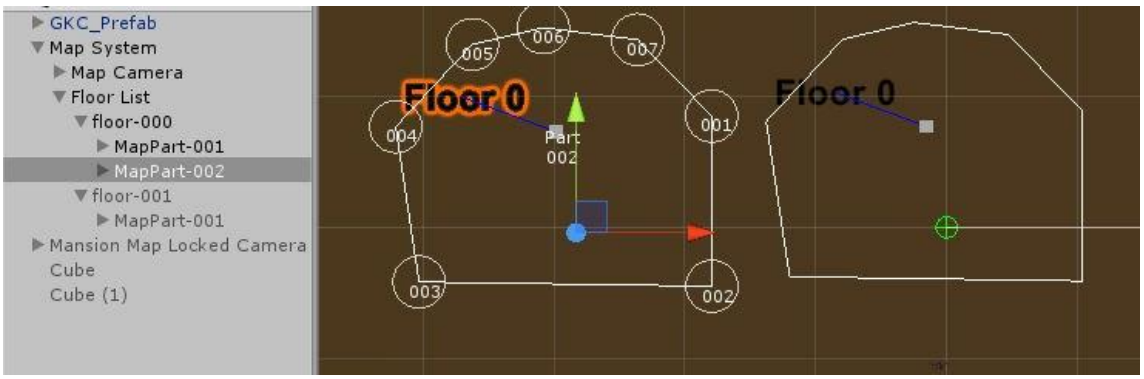


Duplicate Map Part

This is useful to use parts already created of a floor for equal or similar rooms or shapes. For this, go to the map part that you want to copy, Map Tile Builder inspector, and press the button Duplicate Map Part.

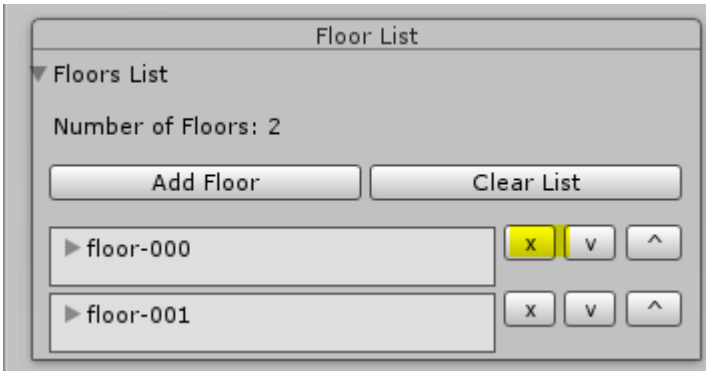


Then, like before, another map part is created, with the same hierarchy that the copied, so you can move it to the place you need. The text meshes are copied too.



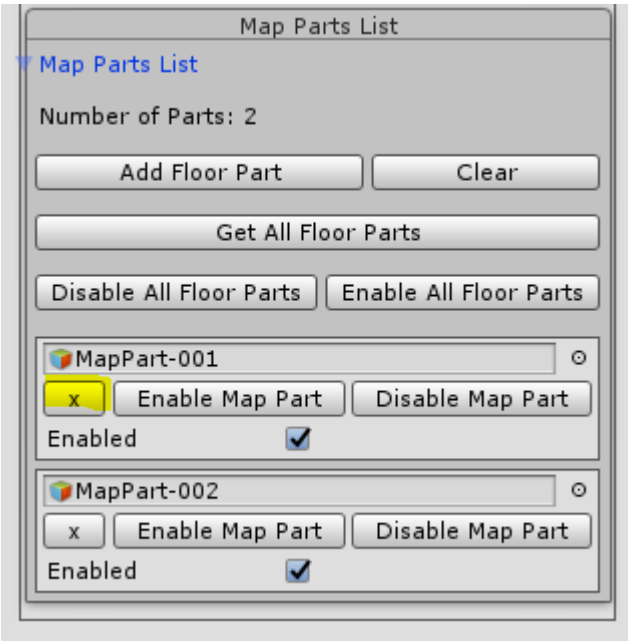
Delete map parts and floors

To remove a map part from a floor, go to the Map Creator inspector and press the button X and both the object in the scene and in the list are remove.



In this case, don't forget to update the floor list in the Map System inspector, like before.

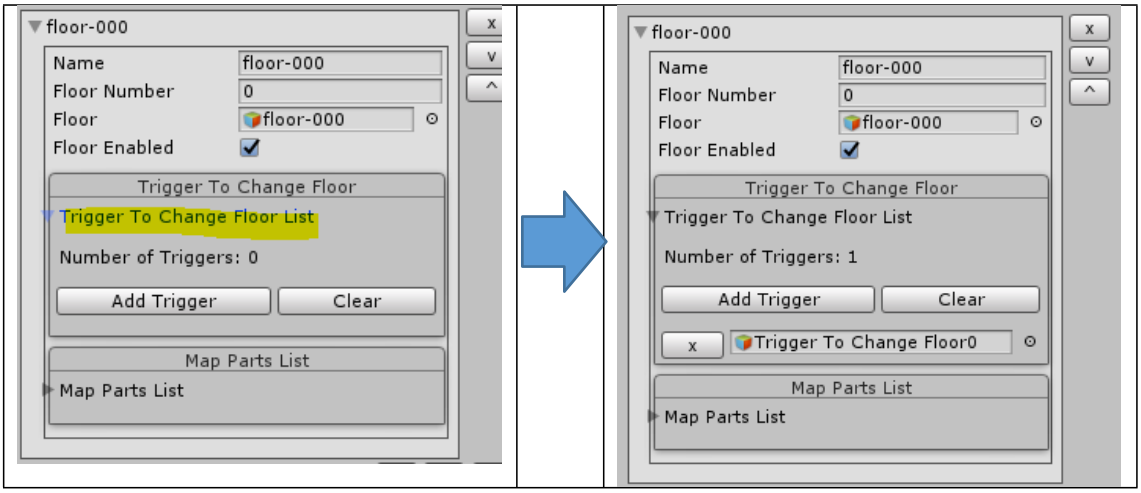
If you want to remove a map part, go inside the floor which has that part and press X button and both the object in the scene and in the list are remove.



Change between floors with triggers

If instead of use vertical distance to know the closest floor where the player is, you can use triggers to change between floors. This can be used better for close spaces levels, instead of opened.

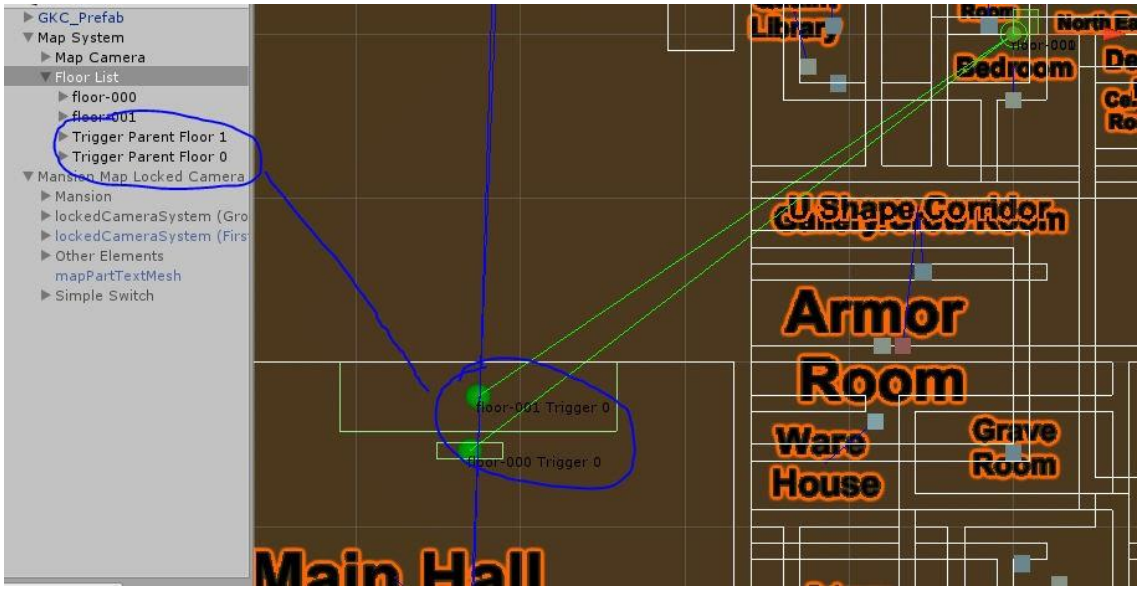
For this, go to the Map Creator inspector and in the floor that you need, open the Trigger To Change Floor list and press the Add Trigger button.

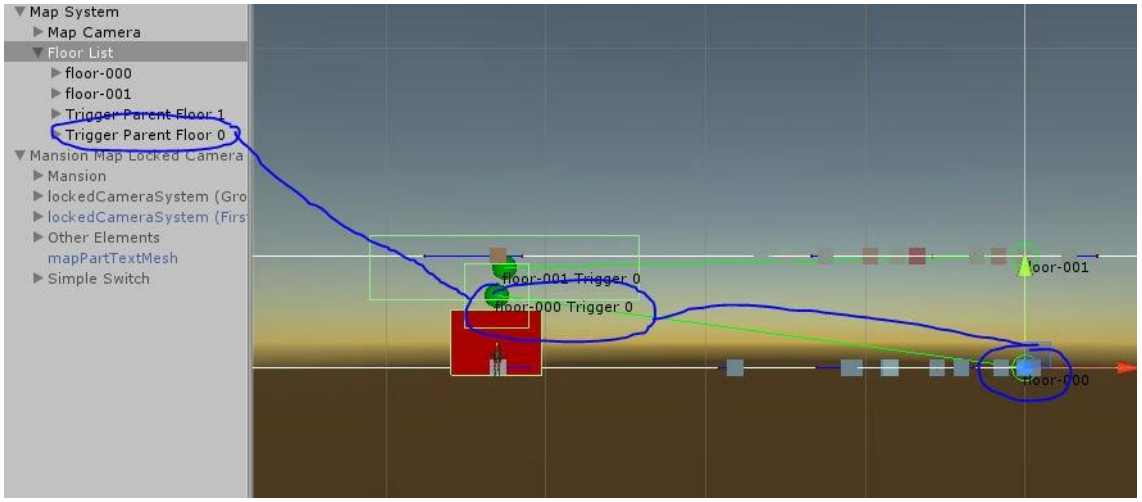


And like before, the gameObjects are created and attached in the proper hierarchy elements.

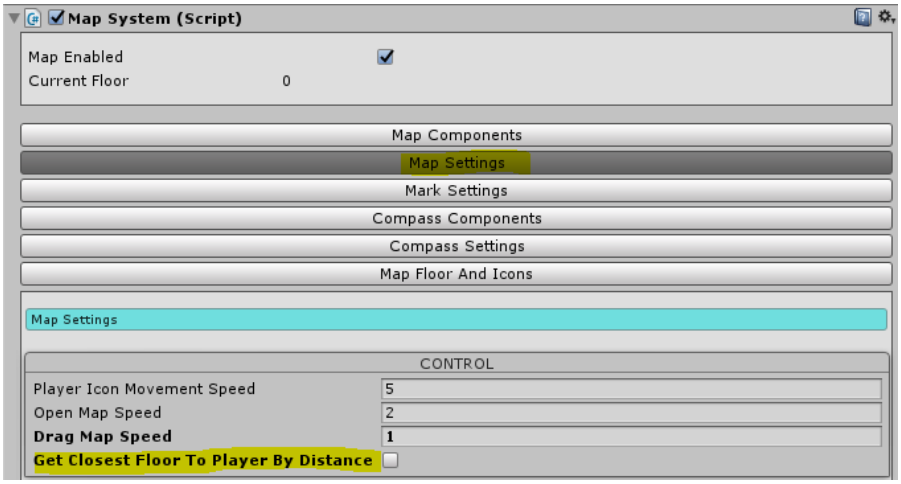
Like this, place this trigger where you want the map floor is changed to the Floor 0. This system is made to used two triggers for zone, it means, one trigger to change to floor 0 and one trigger to change to floor 1 in every place the floors are changed, both of them close (for example, if the building has two stairs, to walk between both floors, there will be 4 triggers, two for every stairs).

This can be appreciated here, in the demo mansion, there is two triggers in the stairs of the hall, one to change to floor 1 and another for the floor 0. Also, gizmo are used to show the link from every trigger to its floor.

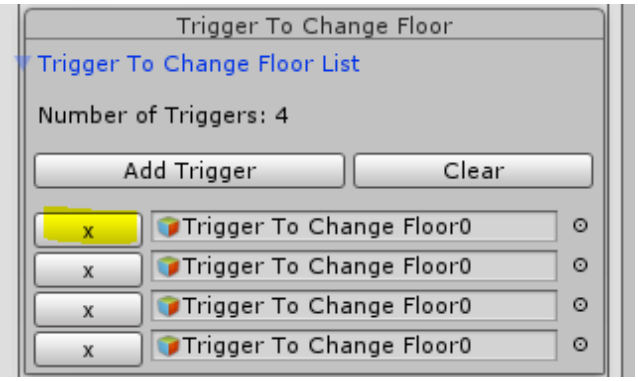




Once the triggers are place, go to the Map System inspector, go to Map Settings and set to false the option Get Closest Floor To Player By Distance.



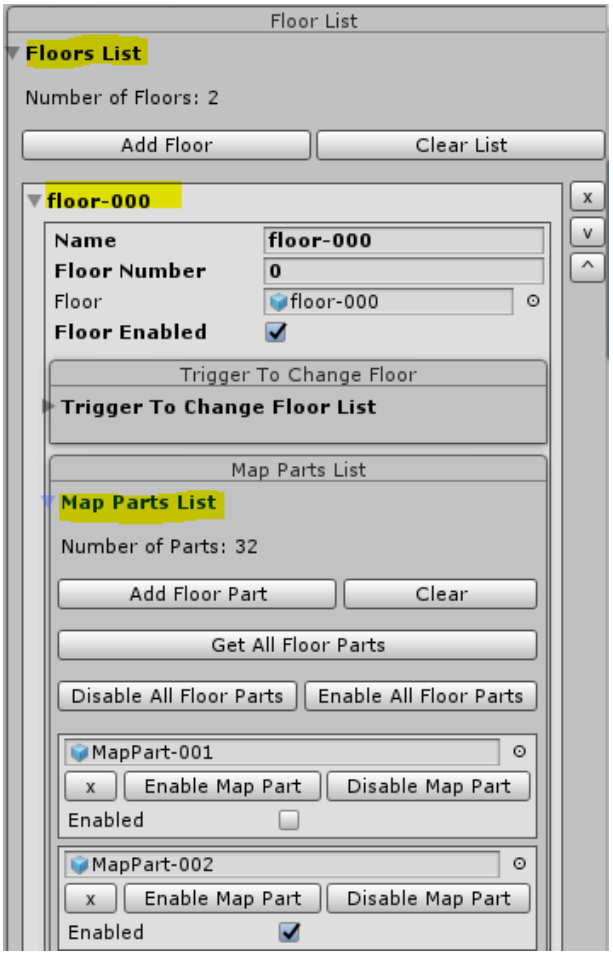
You can add as many triggers as you need for every floor. Also, you can remove any of these triggers with the X button and both the object in the scene and in the list are remove.



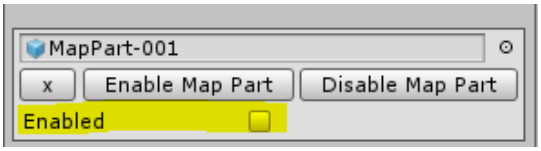
Enable and disable certain parts of the map (map parts or entire floors)

The map can be totally shown in the map or you can select if a certain floor or map part is not shown, so the player needs to reach that zones or get a map pickup of that zone, to make it visible in the map window.

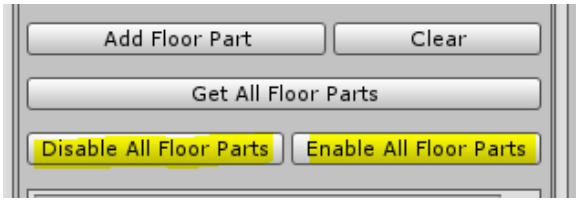
For this, go to Map Creator Inspector and open the Map Parts List of a floor.



Then, you can select every map part state of that floor one by one, using the Enabled option.

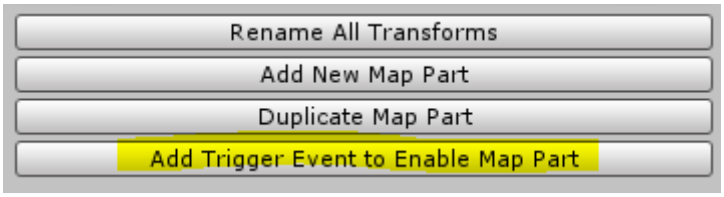


You can also, enable or disable entire floors with the options Disable All Floor Parts and Enable All Floor Parts.

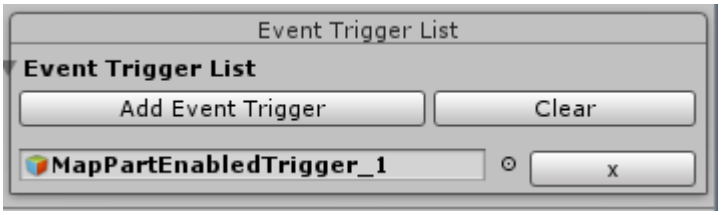


Add triggers for hidden map parts

Once that some map parts have been disabled, you can configure the trigger that will show the map part when the player reaches that zone. For this, go to the map part that you need, Map Tile Builder inspector and press the button Add Trigger Event to Enable Map Part.



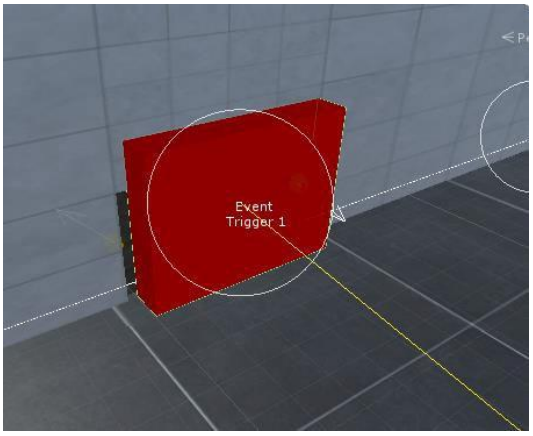
A trigger is automatically created, but you can add as many trigger as you need for the same map part (for example a room with three doors, so it will be activated no matter the door the player uses to enter).



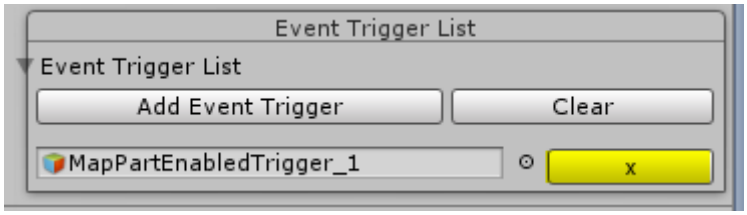
Next, select the MapPartEnabledTrigger_1 in the inspector and the hierarchy will show this element, select it there and place every trigger where you need.

As you can see, in this picture, this trigger is used to show the Main Hall of the mansion when the player enters on it.

Also, the trigger shows this gizmo, making easier to see in the scene.

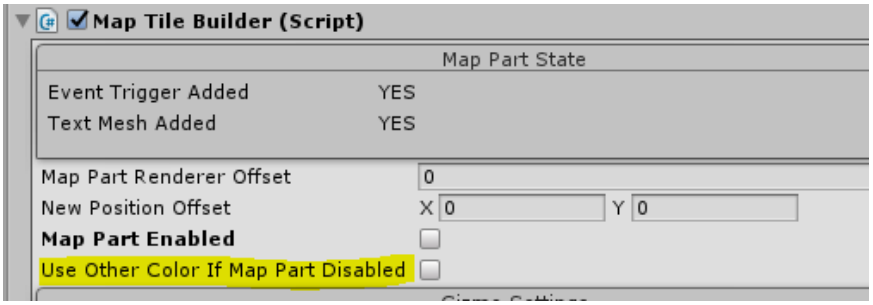


To remove any trigger, press the X button and both the object in the scene and in the list are removed.

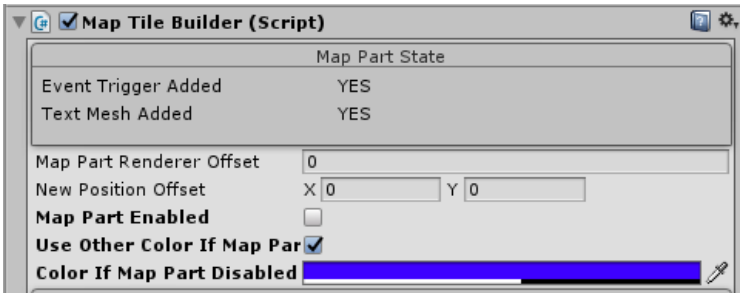


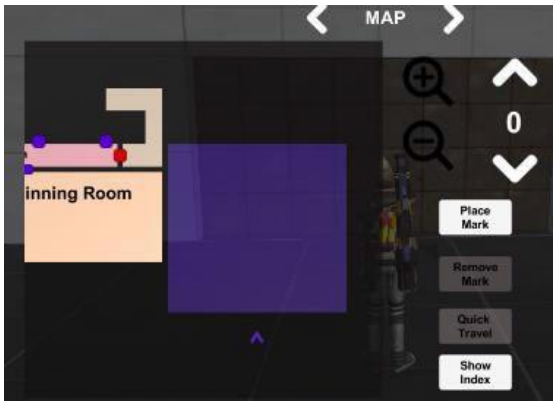
Show a hidden map part with other color until it is visible

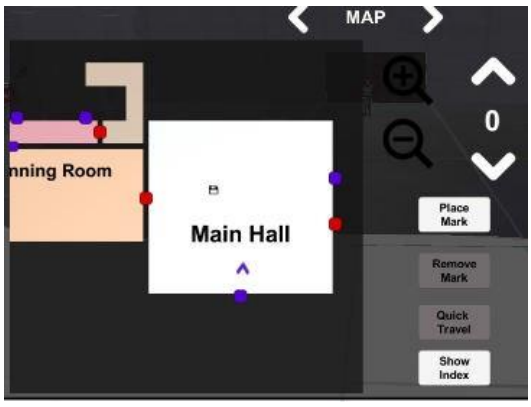
This can be used to show that map part as hidden but showing its shape, but not its text mesh. For this, go to the Map Tile Builder inspector and enable the option Use Other Color If Map Part Disabled (make sure the option Map Part Enabled is false).



In this case, a color field appears to be configured:







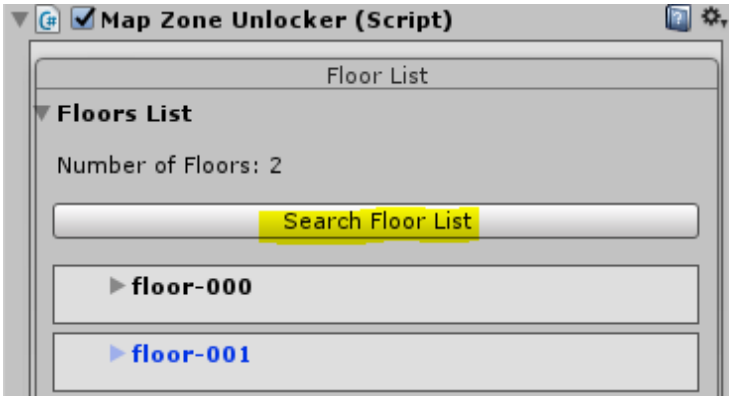
In the above pictures this can be appreciated, in the left the main hall hasn't been found yet, in the right the player has found it with the map part trigger.

Configure map pickup to unlock map parts or floors

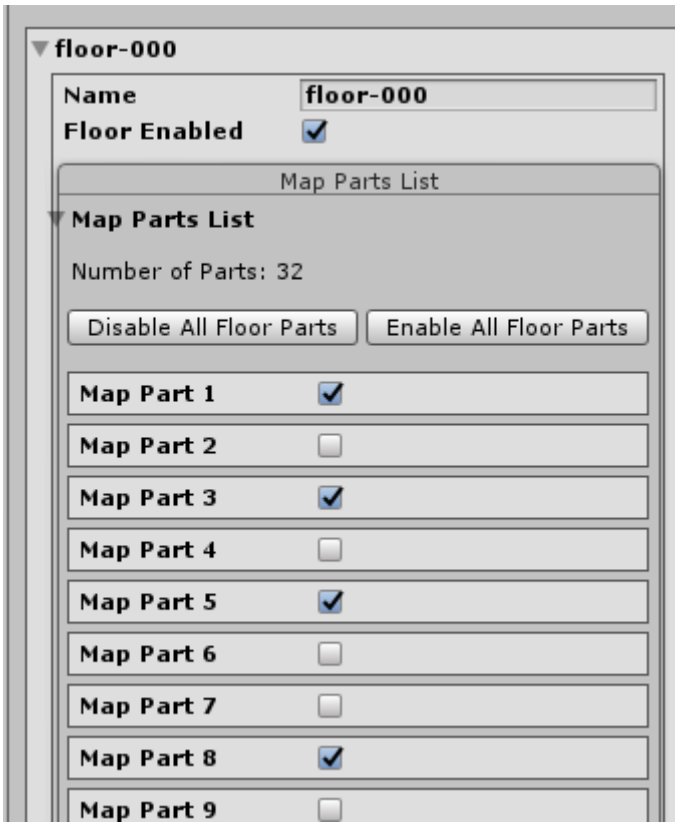
This is useful to hide map parts or floors at the start of the game, so the player can unlock these zone maps with pickups for example.

For this, once that some floors or map parts are disabled, drag and drop the Map inventory prefab, in Prefabs/Inventory/Usable.

In that object, go to the Map Zone Unlocker inspector and press the button Search Floor List to get the configured map floors in the Map Creator.



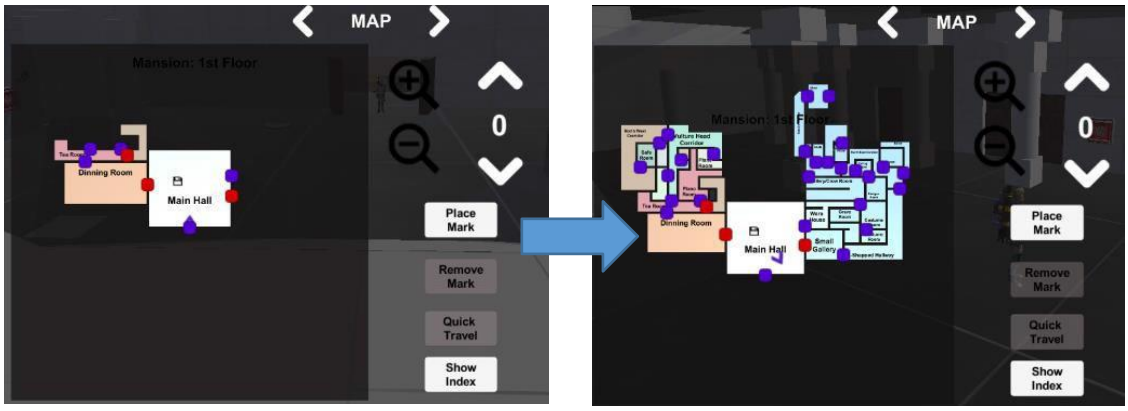
Open every map floor list that you need and select the map parts that this pickup will unlock.



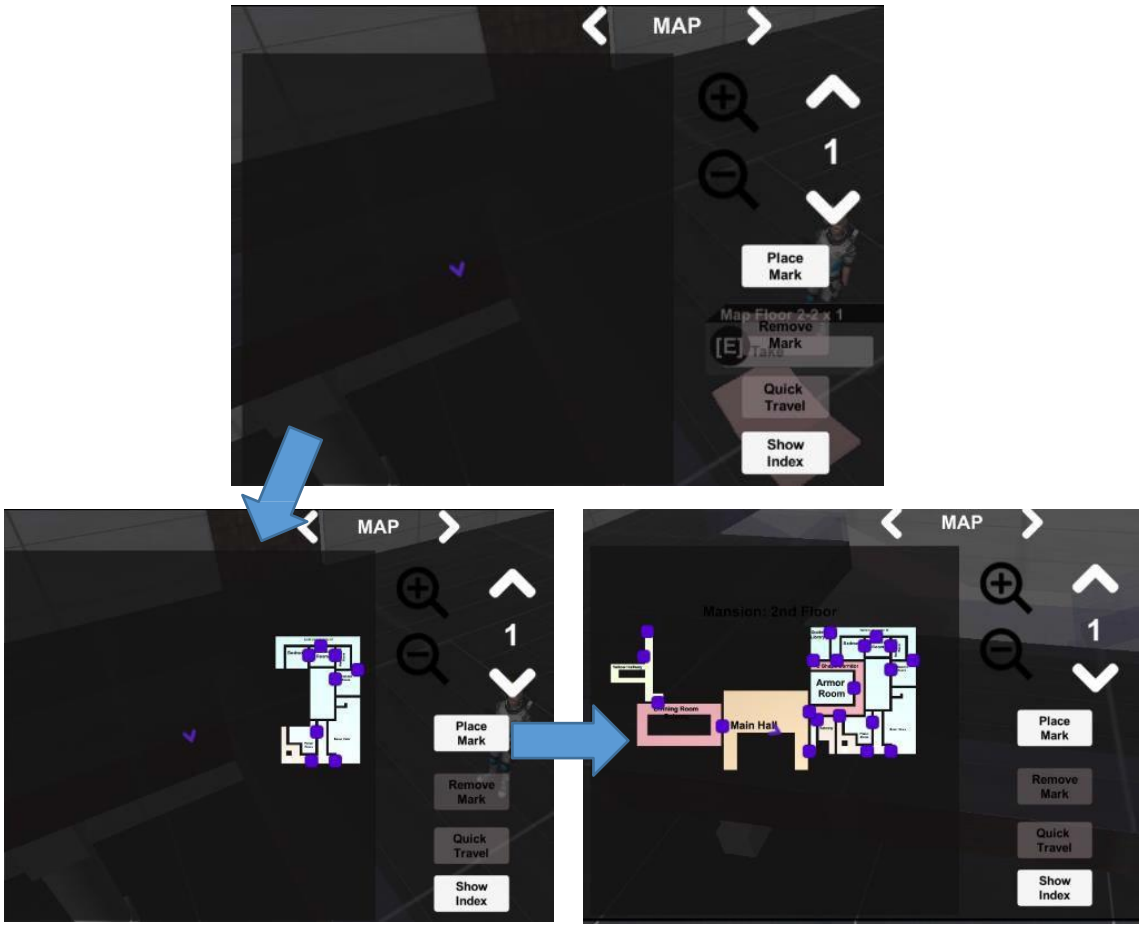
So like this, when the pickup is grabbed by the player, these zones will be shown in the map window.

For example, in the mansion demo, there are three map unlocker pickups, one that enables all the floor 0 and other 2, which every enable half of the map of floor 1.

Floor 0 before and after unlock the map.

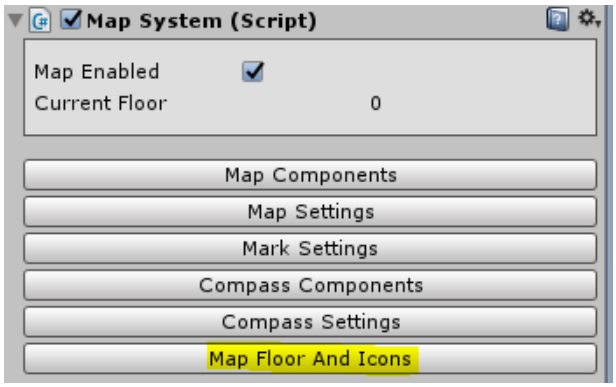


In this case, the first floor map is totally disabled and the player picks one map unlocker and then the other.

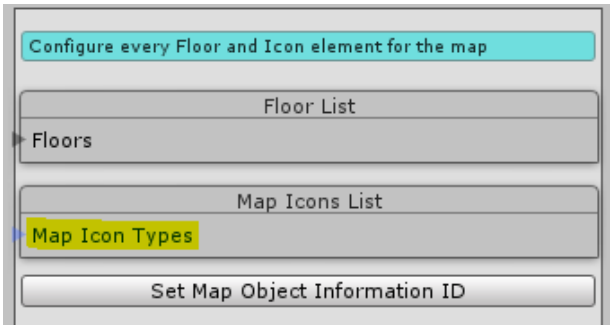


Configure Map object information Icons

In the Map System inspector, you can configure the icons used in the map window (which is assigned in the map object information). To add new ones or configure them, go to the option Map Floor And Icons.

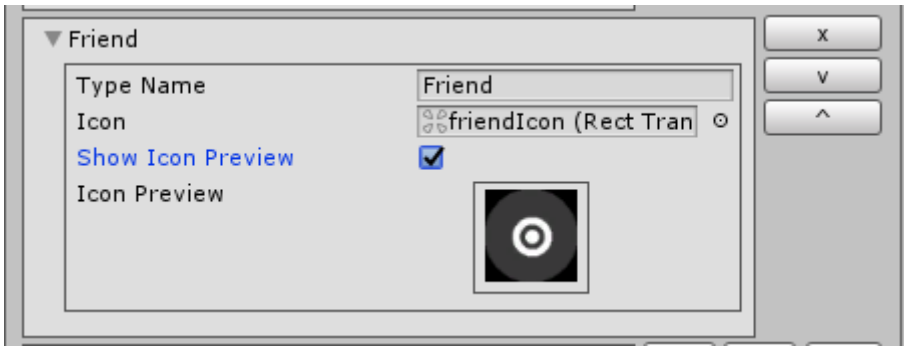


And then, to Map Icon Types.



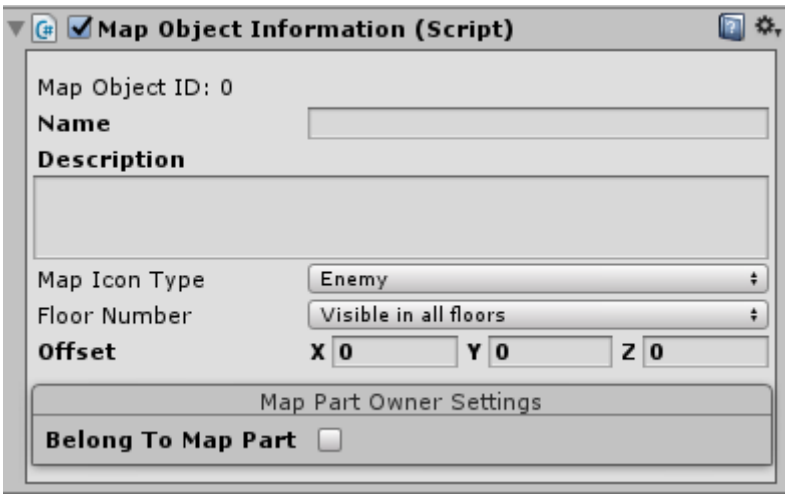
In this list you can add new type of icons:

For this, press the button Add Icon, configure a name, drag and drop one of the other icons already made, set the image you need and make a prefab of it. Configure that prefab in this list.



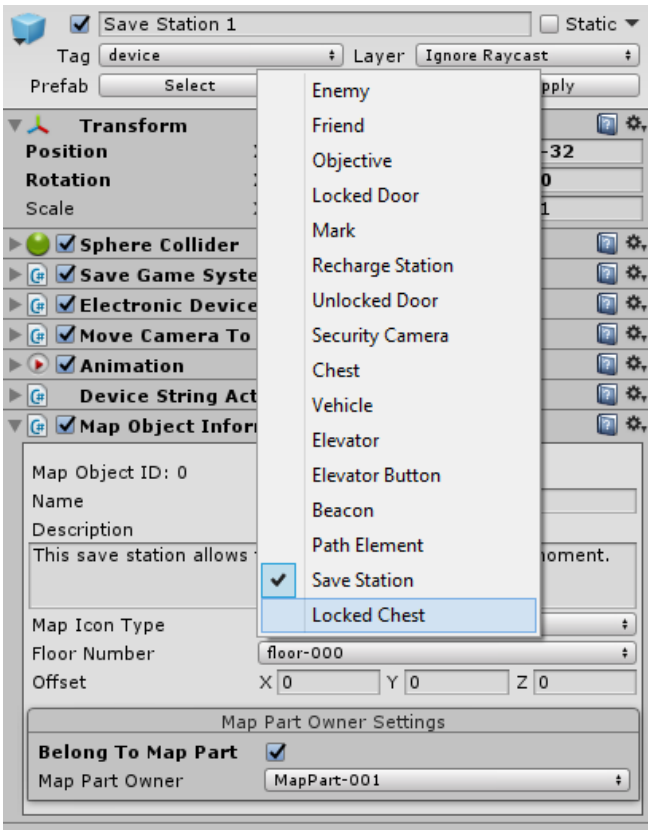
Use Map Object Information

Once that you have a list of icons, go to an object that you want to be shown in the map (an enemy, a door, etc…) and add the component Map Object Information.

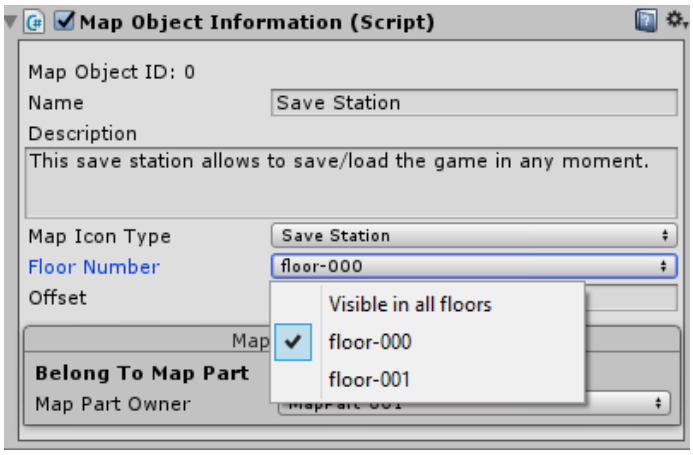


In this component, you can configure a name, a description, the type of icon, the floor where this icon is shown, an offset for the icon, and the map part where the icon belongs, for example a save station inside a room.

The Map Icon Type field will show the same list of icons configured in the Map System previously.

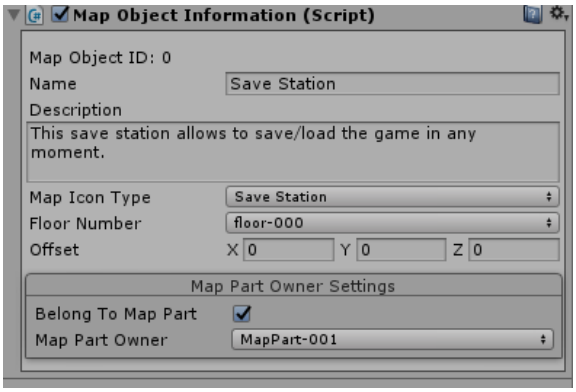


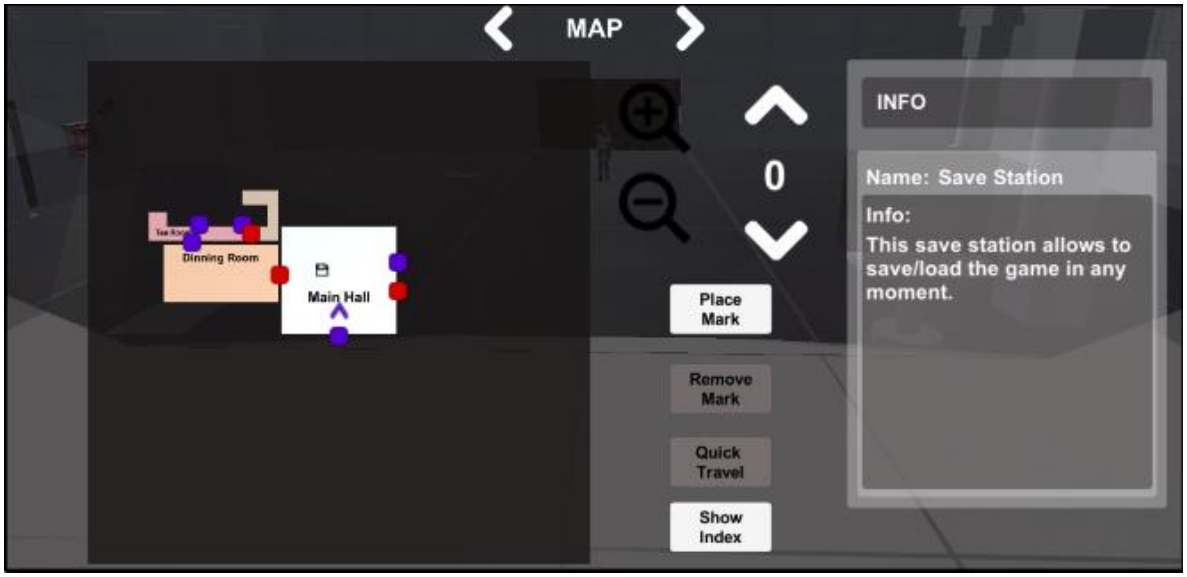
The Floor number field works in the same way, it will show the list of floors configured in the Map System, used to configure in which floor the map icon is shown.



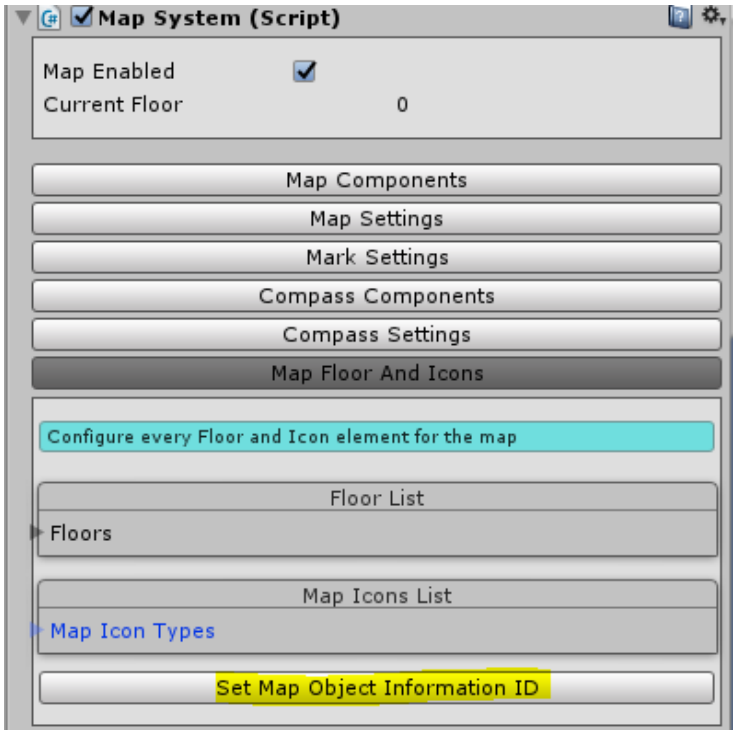
Another field, called Visible in all floors can be selected to show that map icon in every moment in the map window, no matter the current floor where the player is.

For example, a save station with next configuration will show this info in the map window:





Once that you have configured all the Map Object Information components in the scene, go to Map System inspector, and press the button Set Map Object Information ID.

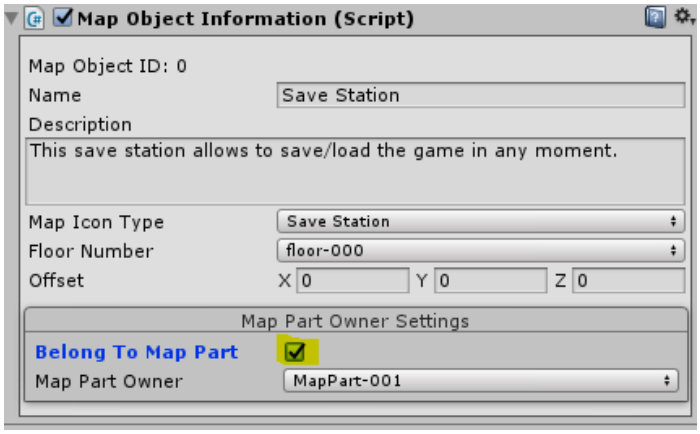


This is used in the Map System to manage the map icons properly.

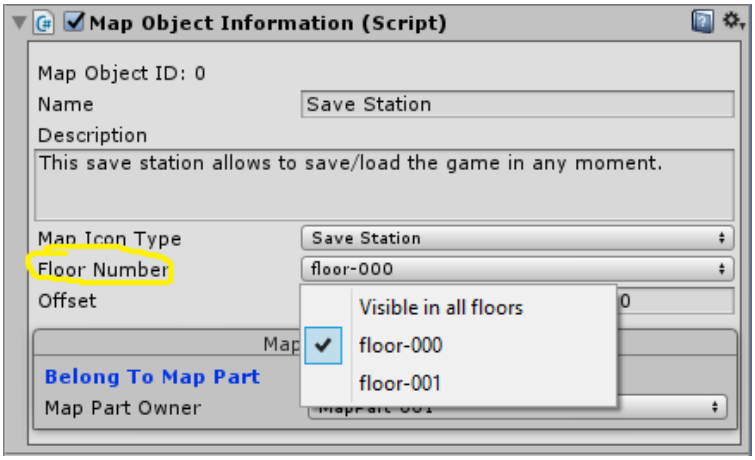
Map object information linked to a map part

This can be used to hide icons in the map on map parts that are hidden, so if a map part is not enabled, the icons inside that map part are disabled too and once the map part is enabled, the icons will be shown in the map as well.

For this, go to the Map Object Information inspector and enable the option Belong To Map Part.

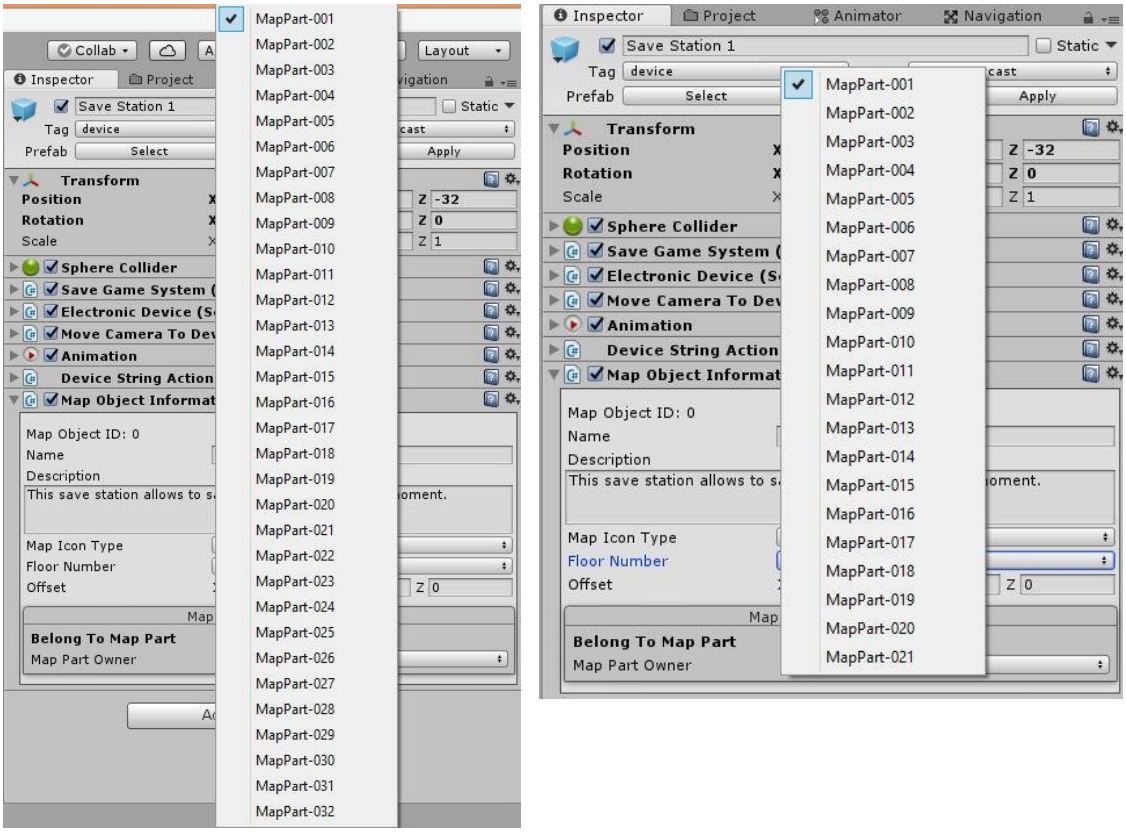


Then, select the Map Part Owner, it means the map part that contains that icon. Use the list of map parts to select where the icon belongs. This list is shown according to the Floor Number configured in the inspector.

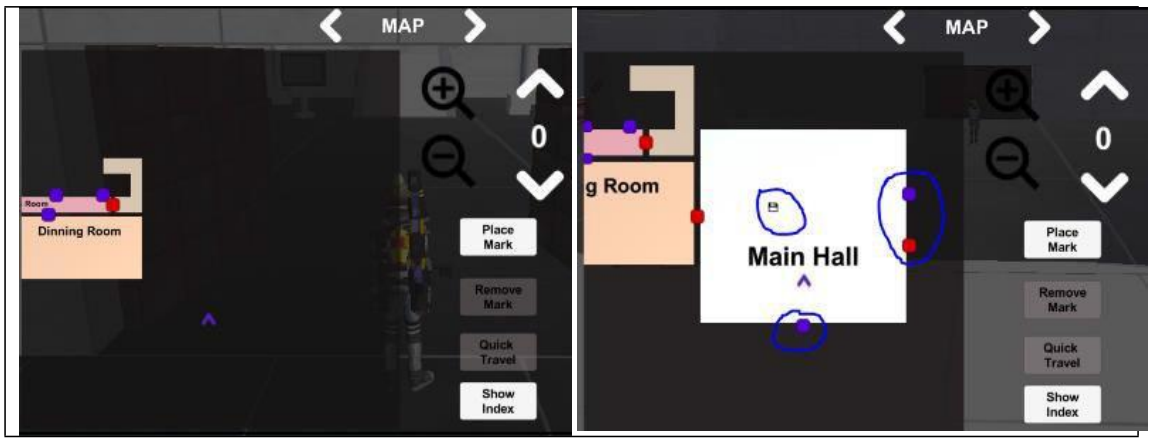


The list of floors are configured automatically from the map floors configured in the Map Creator inspector.

Like this, if you select the floor-000, it will appear the map parts list of that floor. In the left picture the map part list shown belongs to the ground floor. In the right picture, the map part list belongs to the first floor.

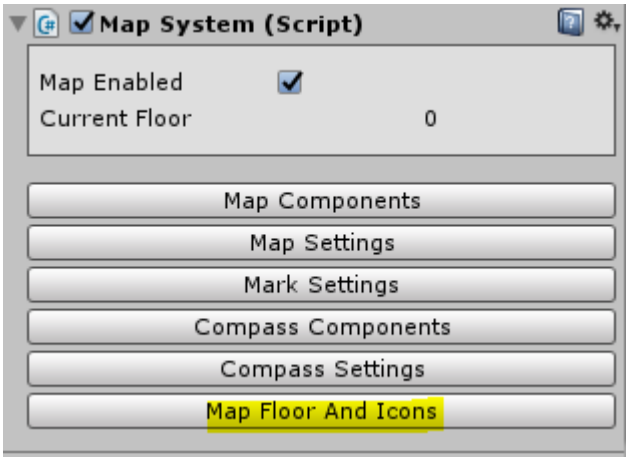


In this example, the Map Object Information is used for a save station in the main hall room, which is disabled until the player enters in the room. Also, there are 3 doors which are configured with the same unlock values.

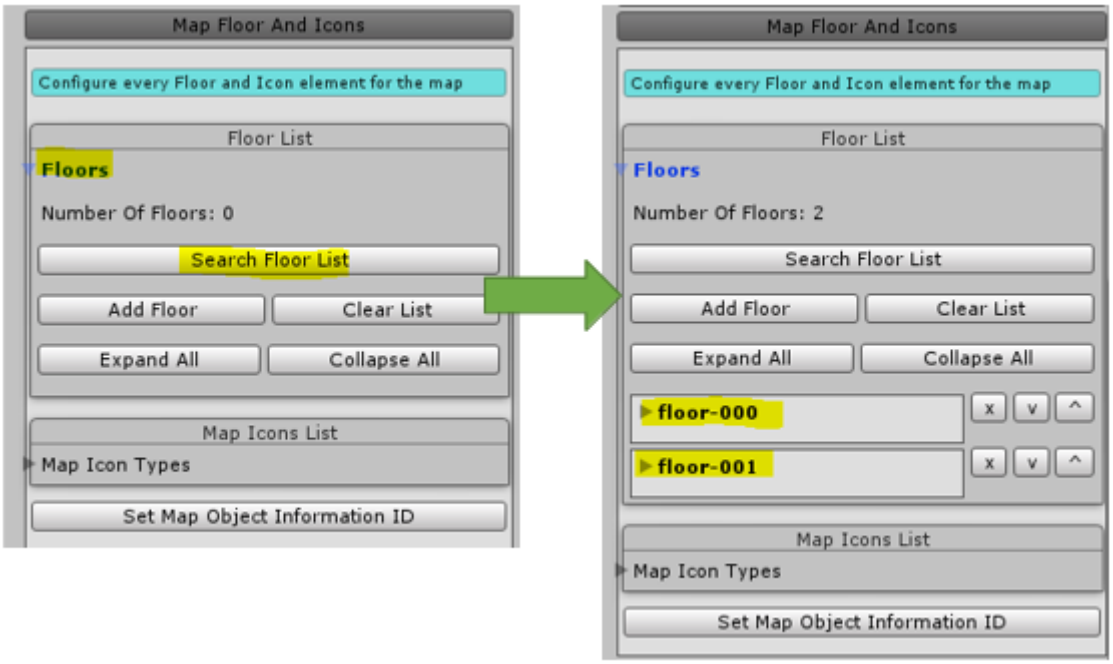


Assign the created floors to the Map System

Once that you have create all the map floors (or if you have added or removed on floor of the list) go to Character gameObject, Map System inspector and press the button Map Floor and Icons.



Go to Floors, and press the button Search Floor List. The floors configured in the Map Creator inspector will be configured automatically.

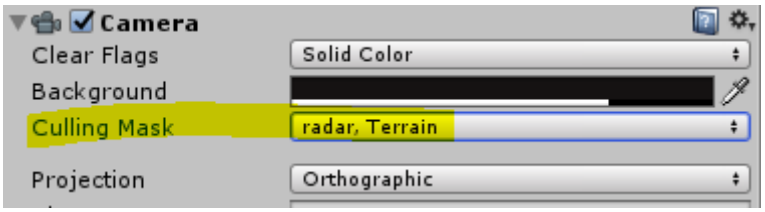


If you don’t need to use the map system, don’t configure the above steps and it will be disabled.

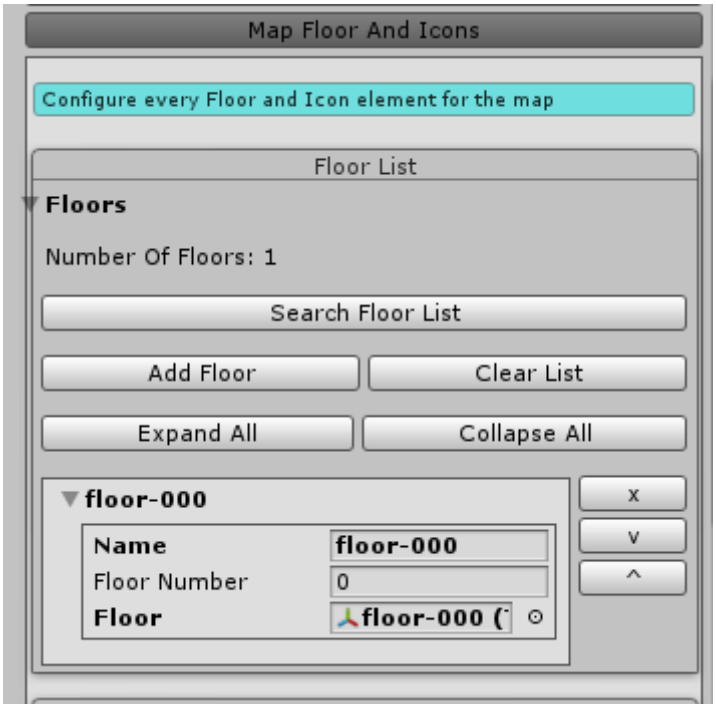
Render terrain in the map window

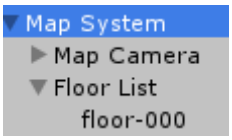
To make the terrain visible in the map system, follow these steps:

- Assign the layer Terrain to the terrain gameObject.
- Inside the Map System gameObject there is a camera, to render the tile meshes created with the map creator. In that camera, add the layer Terrain in the culling mask.

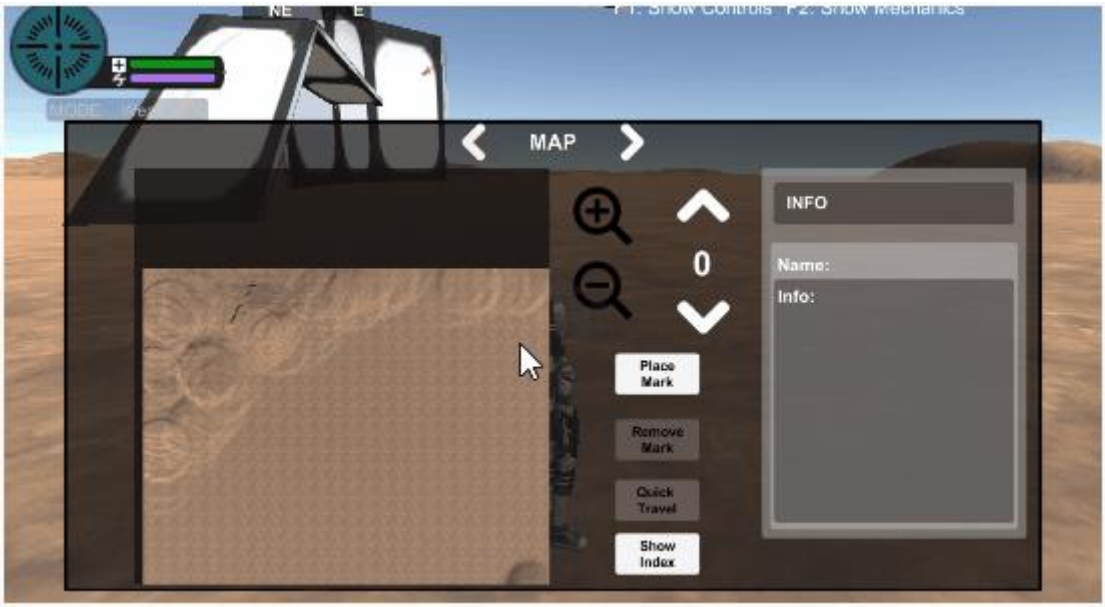


- Make sure to create at least an empty floor in the Map System component in the Character gameObject, it doesn't need to have any map part.





Then, the terrain will be visible in editor mode in the mini window and ingame.



SCREEN OBJECTIVES SYSTEM

PICKUPS

- CRATES
- CHEST
- PICKUP OBJECTS
- PICKUP ICON MANAGER

PLAYER GRAVITY SYSTEM

- GRAVITY TRIGGERS
- CIRCUMNAVIGATE OBJECTS

INVENTORY

Warning: this part of the documentation for the inventory is related for version 2.4b, which is in process and it will be published before mid-March.

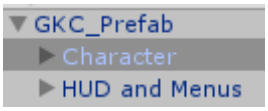
The inventory is composed from this components:

- **Inventory List Manager**, is the general manager used to create the full list of inventory objects used in the game, with the properties which determine if an object can be used, dropped and or equipped. It allows to create the icon used in the inventory menu and create the prefabs automatically.
- **Inventory Manager**, is the manager for the player, used for the inventory menu, where the player can use, drop and equip (this option will come soon) every object. It allows to select the objects configured in the Inventory List Manager.
- **Pickup object/Inventory Object**, these two are the components used for pickups, so when the player grab them, these are stored in the inventory manager, allowing to the player to use them as he needs.
- **Use Inventory Object**, used to configure the elements needed from inventory in a part of the level, like for example a locked door which needs a key or a fuse box which needs some fuses and a wire. It allows to configure 1 or more elements to place in the “puzzle” to solve. Here is also configure the object or objects and the functions that are called when all the objects needed are used.
- **Inventory Capture Manager**, it allows to create icons for every inventory object using a rendered image with the mesh of the inventory object. That icon is used in the grid of objects in the inventory menu.

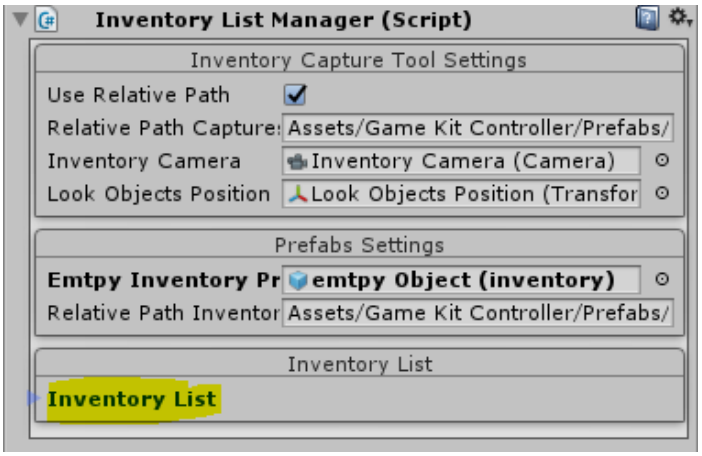
Create new inventory object

To create a new inventory object follow these steps:

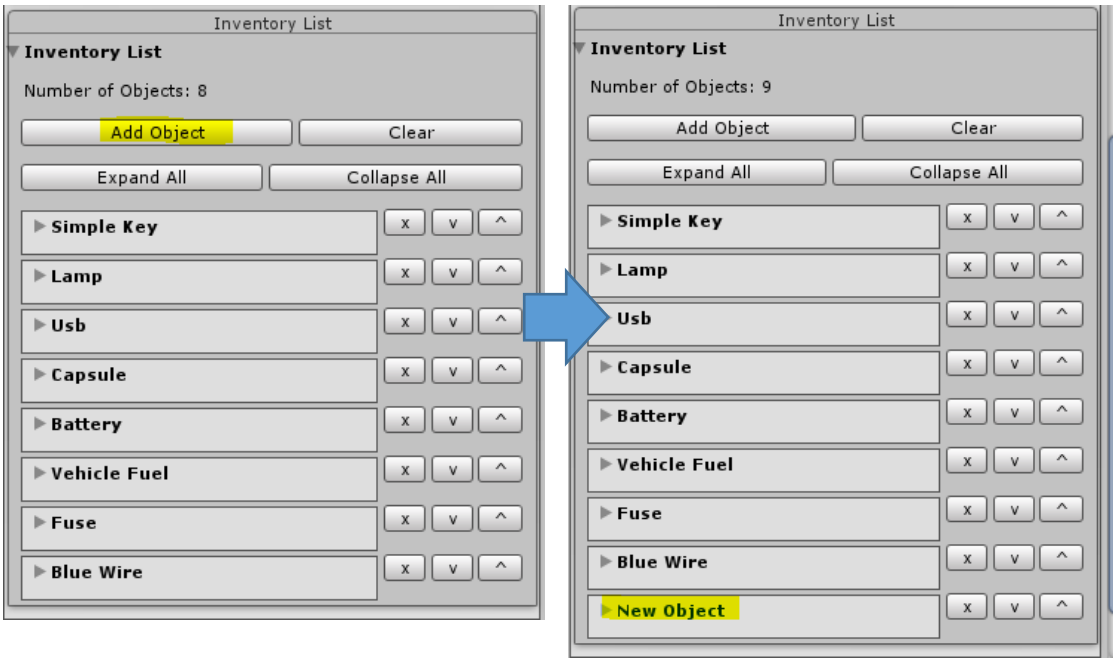
- Select the Character gameObject in the GKC_Prefab.



- Go to Inventory List Manager inspector and open the inventory list.

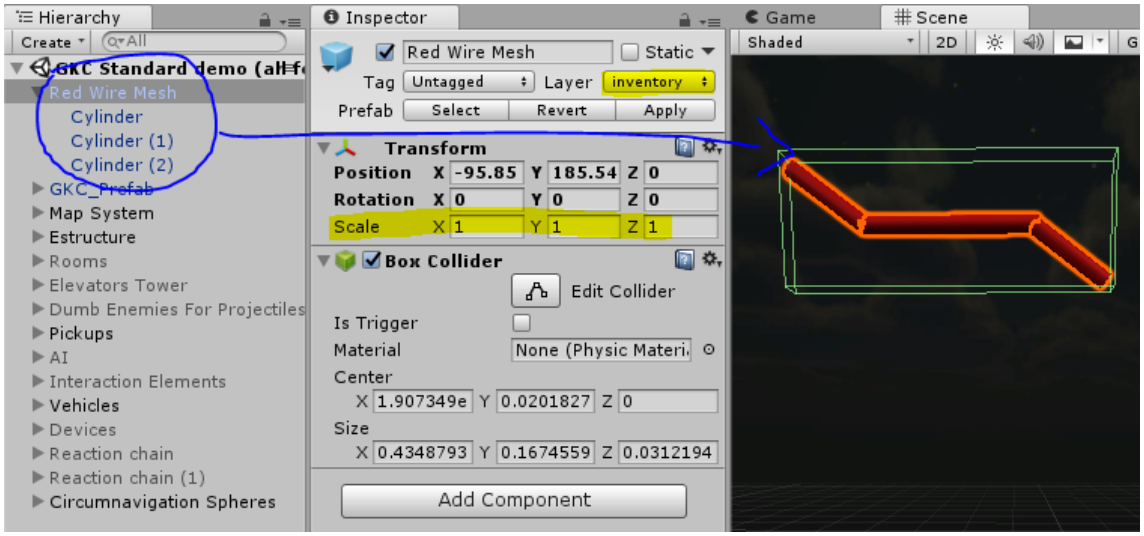


- Press the button Add Object.

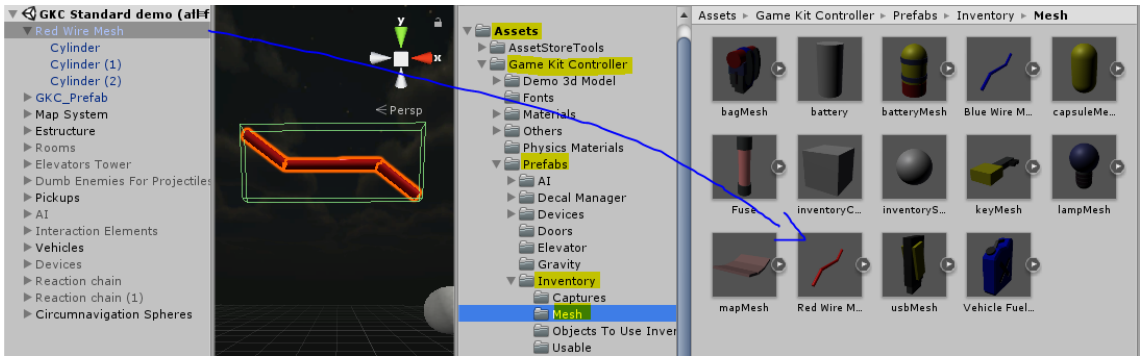


- Open the new object and configure the Name and description.

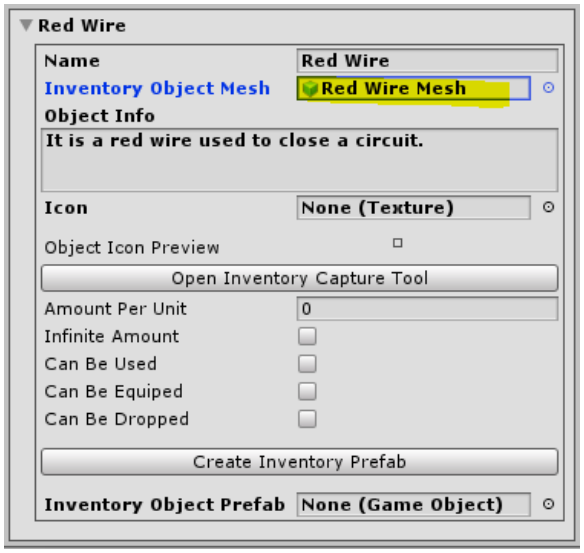
- Then, create the mesh used for that inventory object. It needs to have these settings:
  - The mesh or meshes used in the object are placed in an empty parent, with scale (1,1,1).
  - That parent needs to use the layer inventory.
  - That parent needs a collider adjusted to the object shape and size. You can add as many collider (box, sphere, capsule) as you need if the objects has a complicated shape, but all these colliders need to be configured in the parent.
  - This is not necessary, but the name for the prefab can be the name of the object with Mesh and the end.
- For example, this is the red wire created.



- Once it is created, drop it to the prefab folder in the path Assets/Game Kit Controller/Prefabs/Inventory/Mesh.

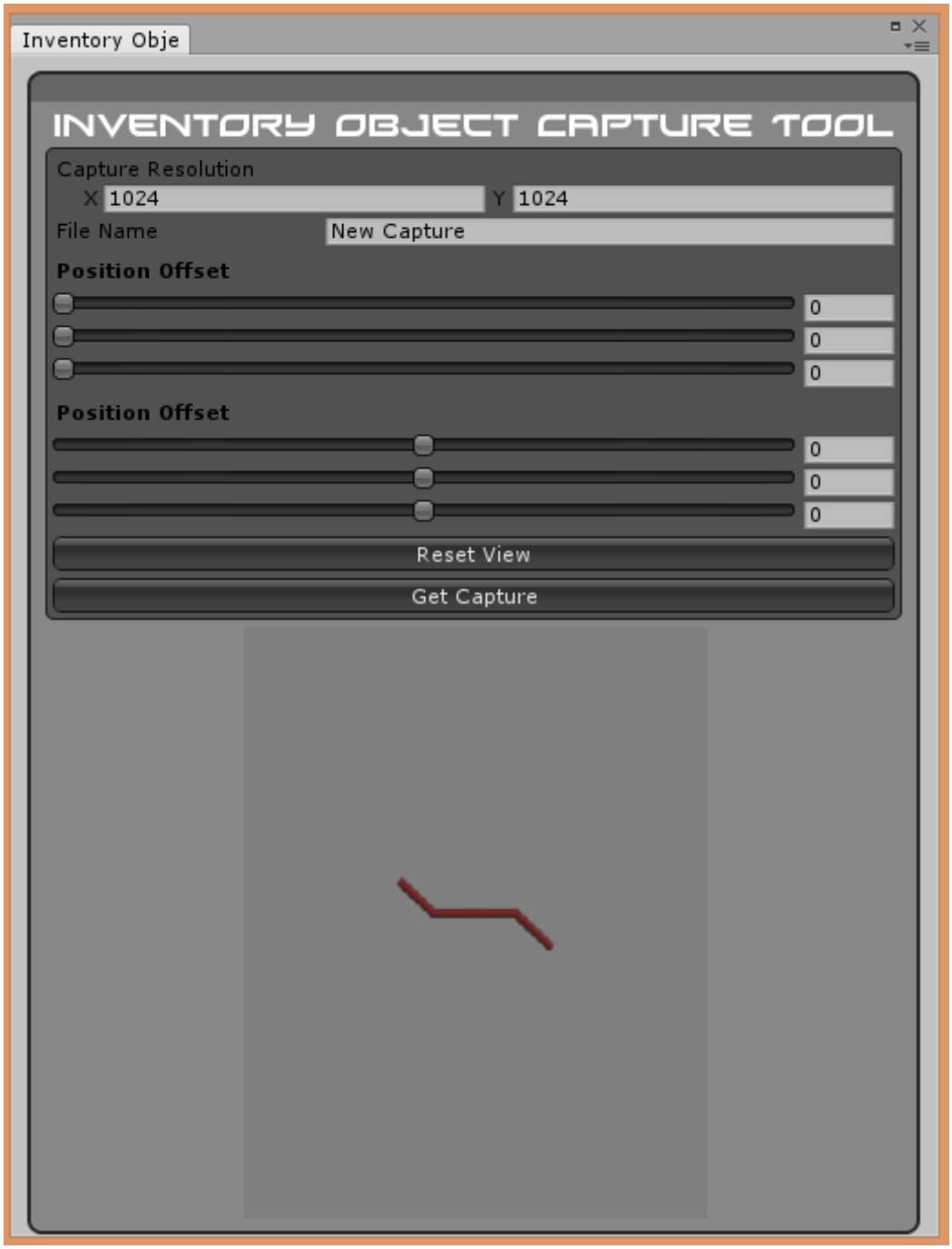


- Go back to the Inventory List Manager, and set the mesh of the new object with the one created previously in the field Inventory Object Mesh.

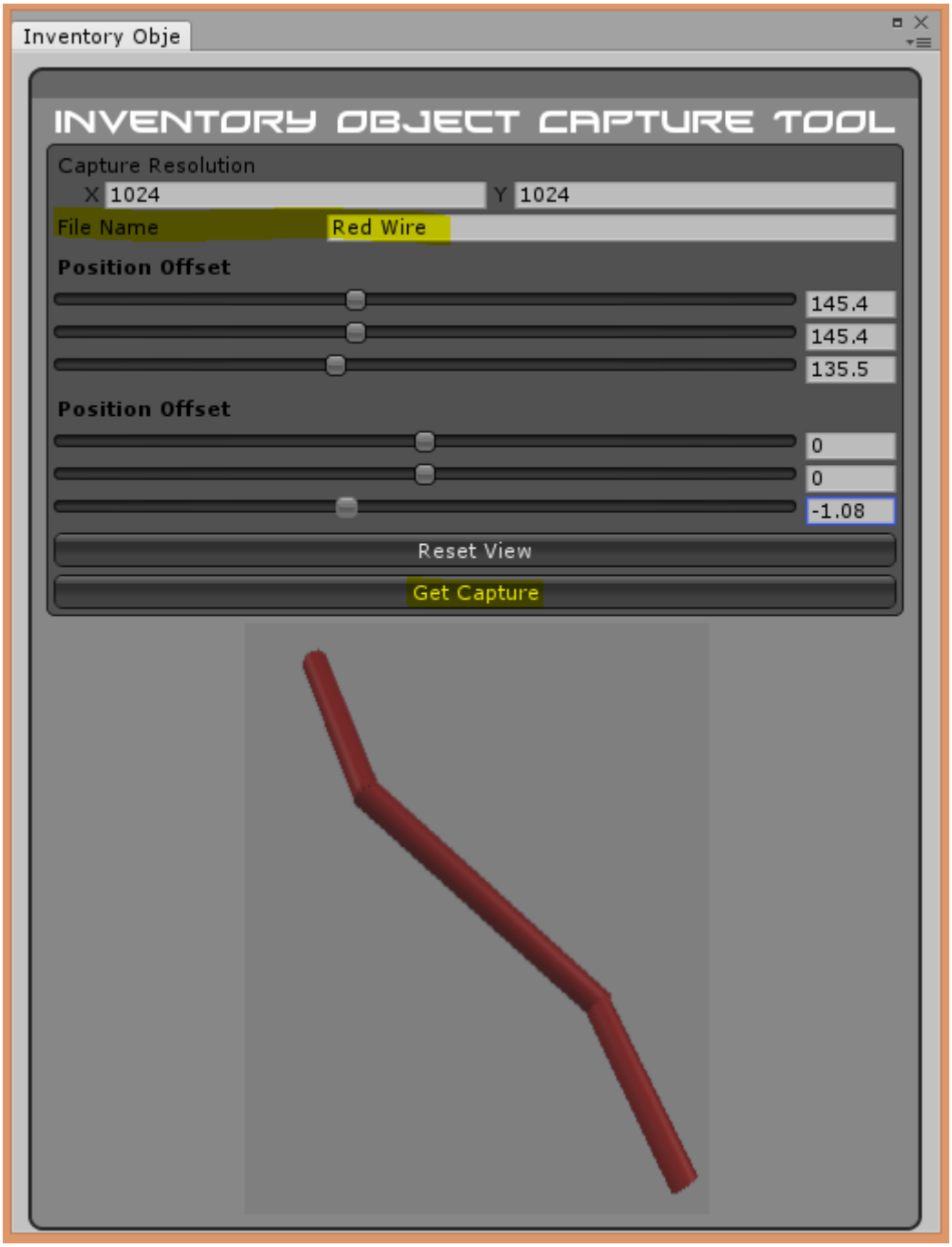


Create inventory icon

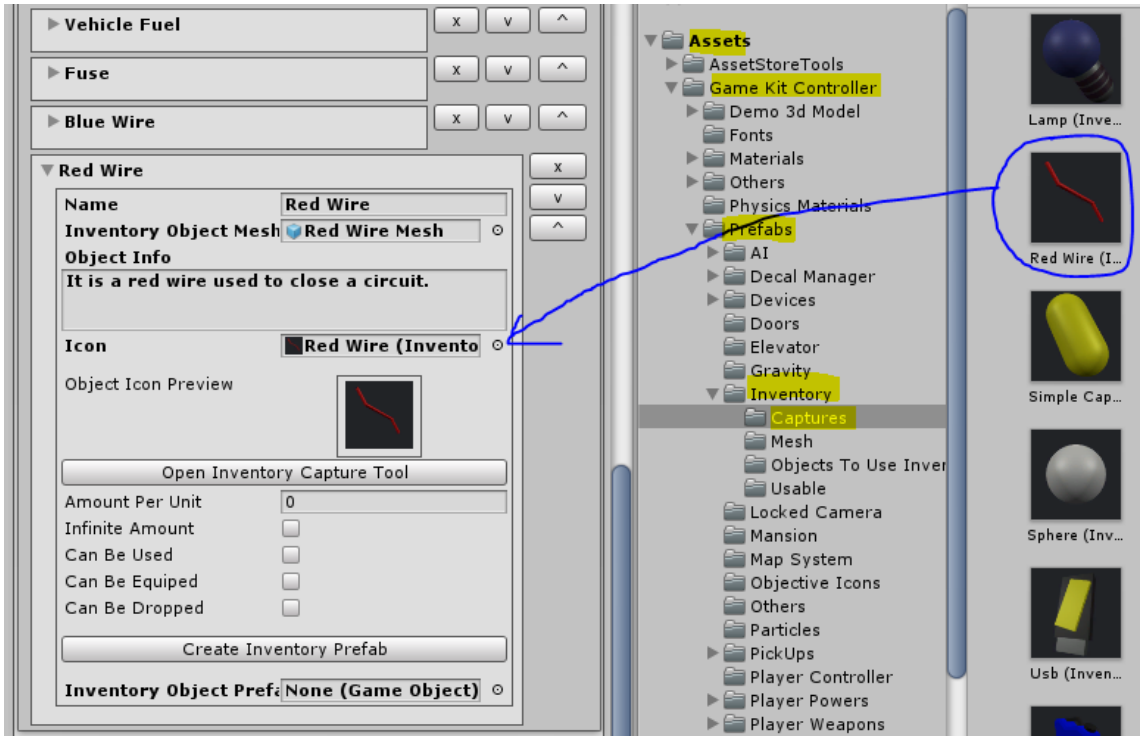
- Now, press the button Open Inventory Capture Tool, so the mesh created is rendered in the window to take a picture of it.



- Configure the resolution of the image (the default value is 1024x1024) and the name of the capture. Also, you can use the values of position and rotation to rotate the object for a better perspective and move its position to place it correctly inside the capture. Once it is configured, press the button Get Capture.



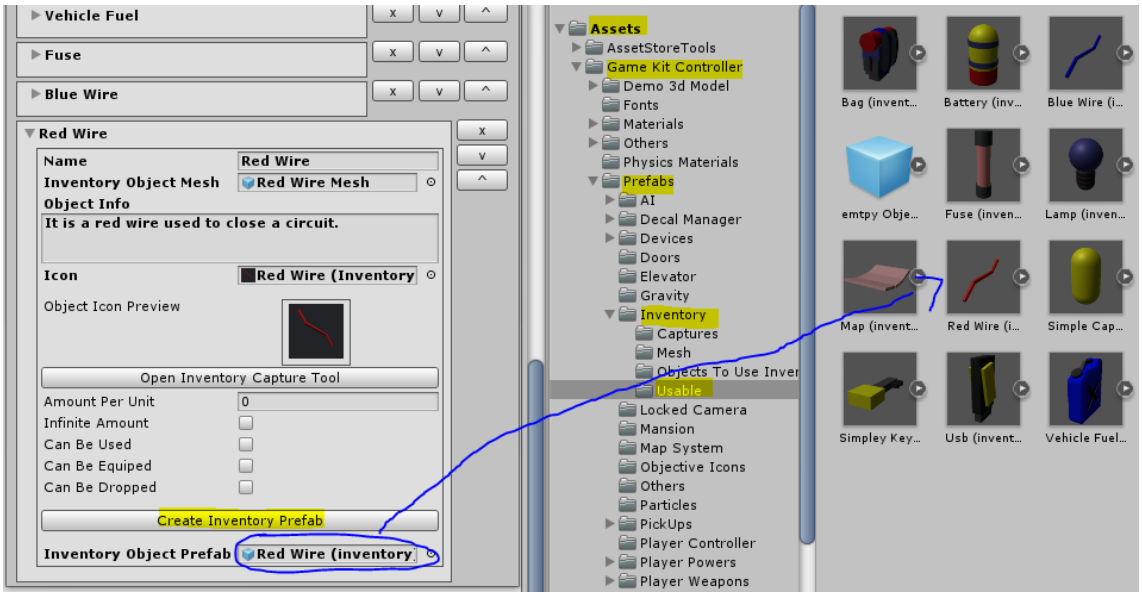
- In the folder path Assets/Game Kit Controller/Prefabs/Inventory/Captures, the new icon is stored. Also, the icon is assigned in the new inventory object in the Inventory List manager.



- Then, configure the values for the object if can be used, equipped and/or dropped. The option Amount Per Unit is used normally for elements like fuel, where the player has for example 10 cans and every can has 10 liters. The Infinite Amount is used to set the inventory object with infinite uses.

Create inventory prefab

- Finally, press the button Create Inventory Prefab to create the prefab of this object, which is the pickup ready to be dragged and dropped into the scene to be used by the player.

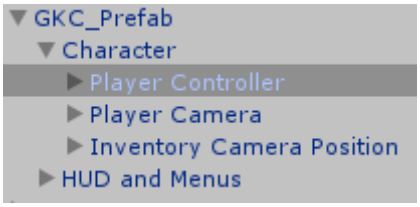


- In this final step, the prefab is stored in the path Assets/Game Kit Controller/Prefabs/Inventory/Usable. Also, this prefab is assigned automatically in the new inventory object created in the Inventory List Manager.

Assign new inventory objects in player's inventory manager

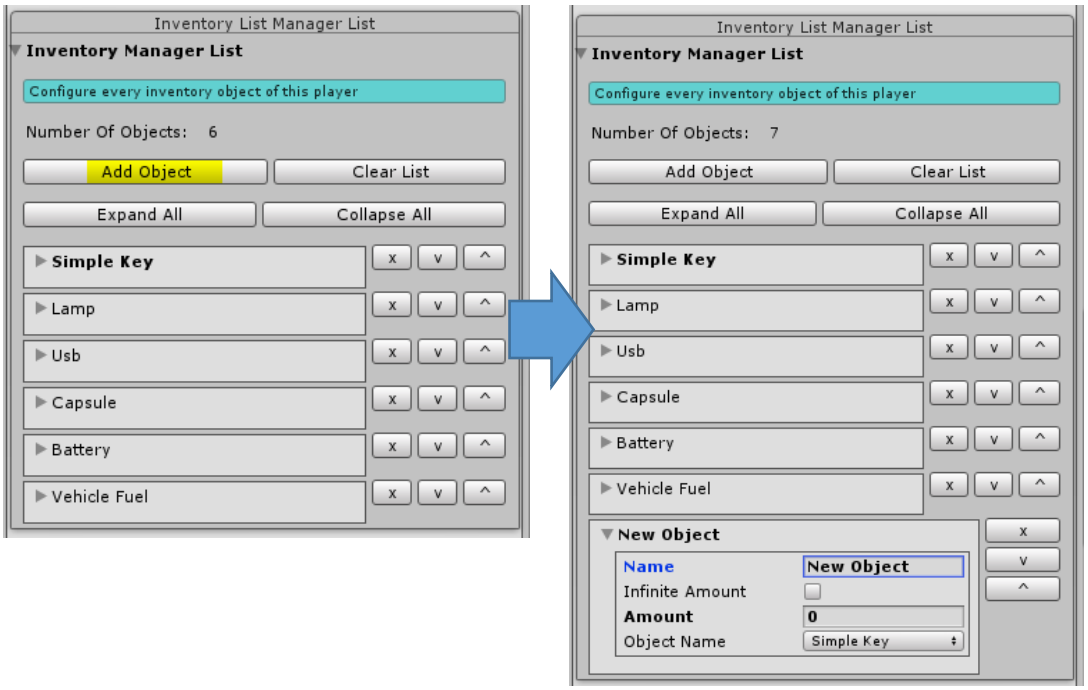
Once the new inventory is created, to be usable by the player follow these steps:

- Go to Player Controller gameObject.



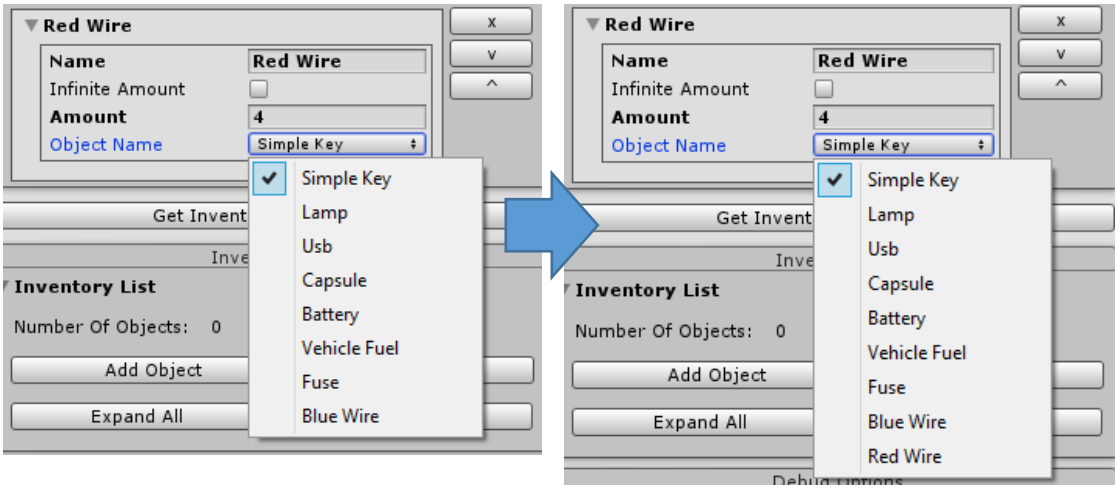
- Go to the inspector Inventory Manager and open the Inventory Manager List.

- Press the button Add object.

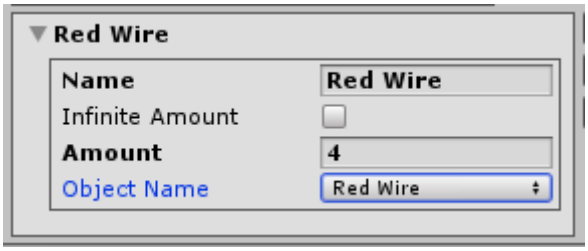


- Configure a name for the object that you want to add from the Inventory List Manager and the amount.
- Press the button Get Inventory Manager List, to get all the inventory objects configured in the Inventory List Manager (this need to be done every time a new inventory object has been added if you want to add it to the player's inventory).

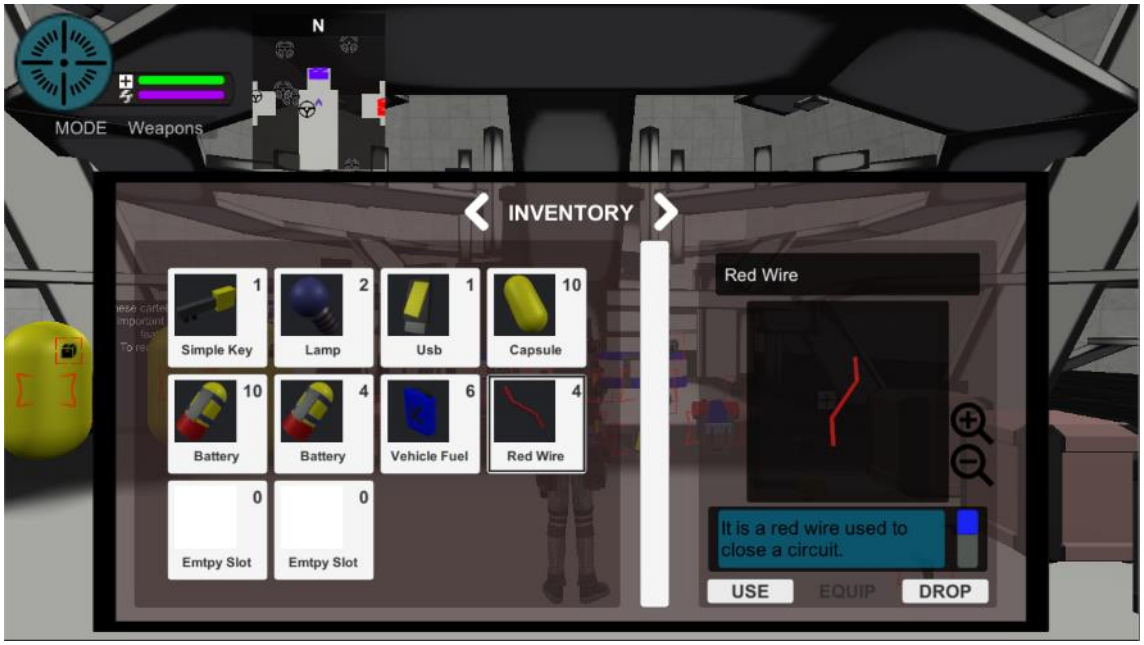
- You can check the difference between the list show in the field Object Name before and after.



- Finally, in the field Object Name, select the new object created in the Inventory List Manager (or the one you need from the previous elements in that list).



To check if everything works, press play an open the inventory menu (by default is the I key). The new inventory object (in this case, the red wire) is shown in the grid and it contains the information configured, the amount, it can be used and dropped.

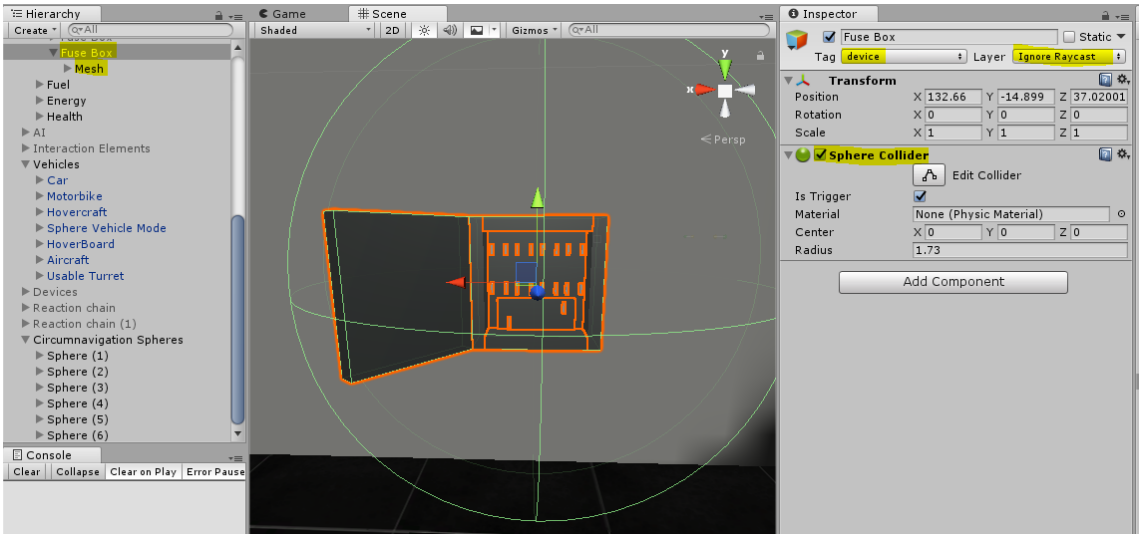


Use inventory objects

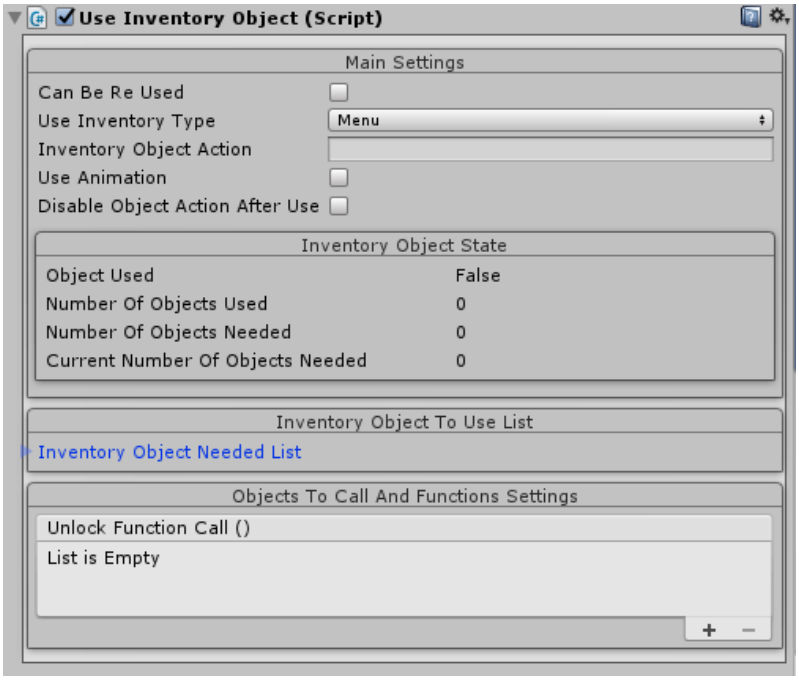
Important: in this explanation, the fuse box is used as example, but this system can be used to create any type of object where to use inventory objects.

Finally, to use the objects created and configure in the previous steps, follow these steps (in this explanation, the fuse box will be used as example to unlock a door and that the mesh for the fuse box is already created):

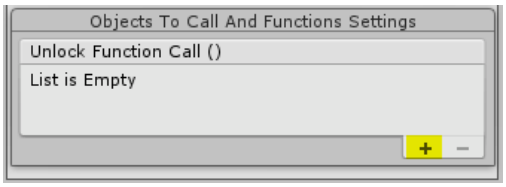
- The parent that contains the fuse box is an empty transform with this configuration:
  - The tag assigned is device.
  - The layer assigned is Ignore Raycast.
  - A sphere collider with the trigger option enabled, used to detect if the player is close enough to the fuse box to use the inventory objects on it.
- So, it looks like this:

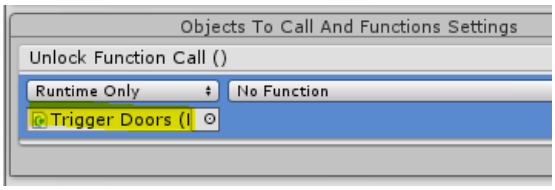


- Add a Use Inventory Object component to the parent.

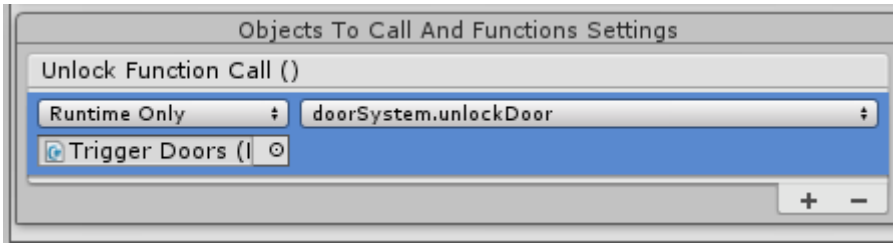


- In the Main Settings, set to true the option Disable Object Action After Use. This is used to disable the Device String Action component icon, which will be added after.
- Also, in the option Use Inventory Type, you can configure how the objects used in the fuse box are placed:
    - Menu, the player needs to open, select and use the object.
    - Button, the player only needs to press the interaction button to use the necessary inventory objects (in case he has them, else a message is shown in the screen to say that the player hasn't the needed objects).
    - Automatic, the player only needs to enter in the trigger to use the inventory objects in the fuse box.
- Add an event to the list of of Unlock Function Call and add the object to use, in this case, add the door to unlock.

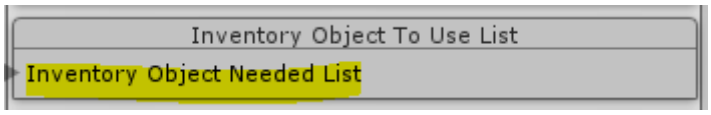




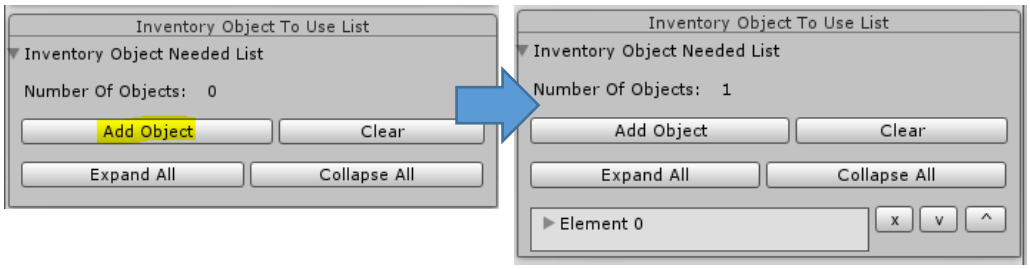
- Add in the Function field, set the UnlockDoor function, from the component Door System.



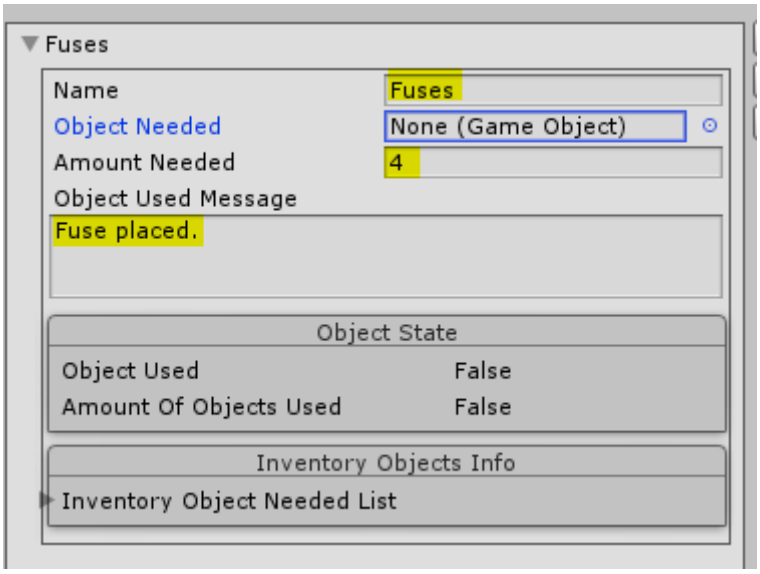
- Now, configure the objects to be used from the inventory in this fuse box. For this follow these steps:
    - Open the Inventory Object To Use Lis.



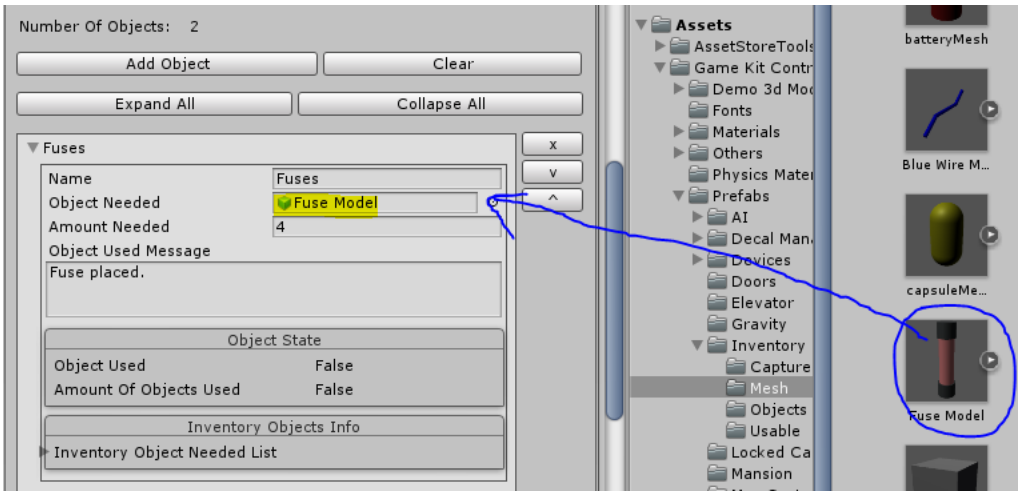
    - Press the button Add Object.



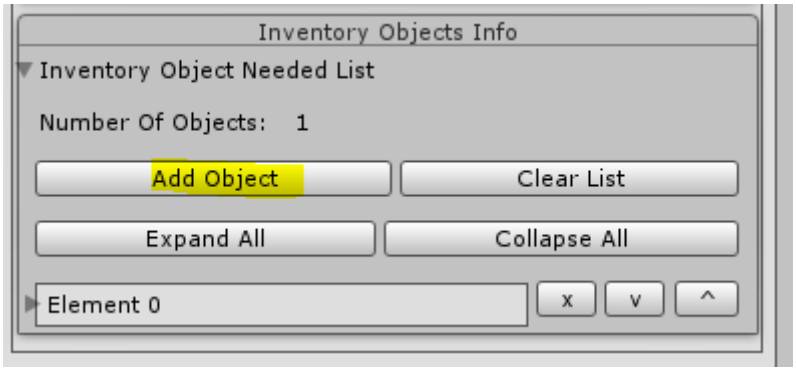
    - Open the new element and configure the values required, name, the amount needed and the message shown in the screen when the object is used.



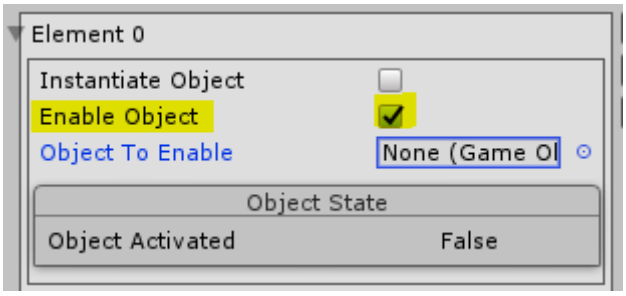
- Then, in the field Object Needed, drag and drop the mesh of the inventory object which needs to be used here, in this case, the fuse. The path of these meshes is Assets/Game Kit Controller/Prefabs/Inventory/Mesh



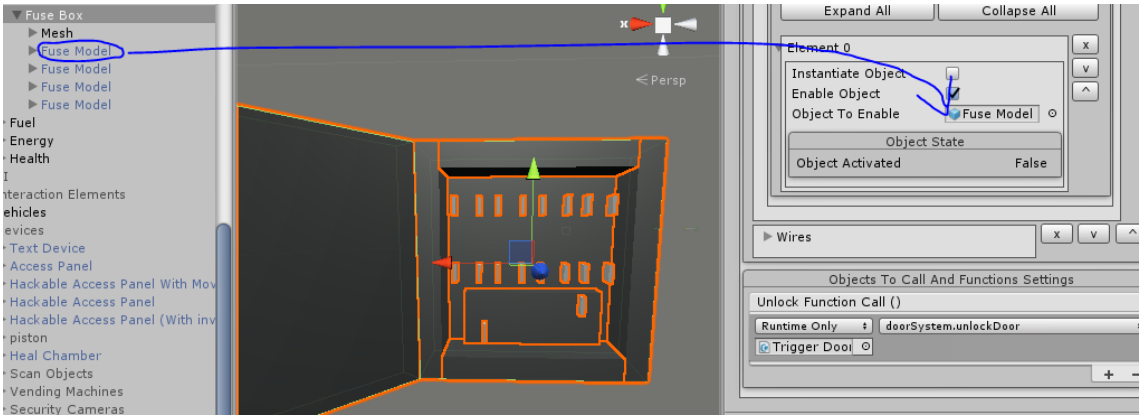
- Open the Inventory Object Needed List inside the current element that is being configured and press the button Add Object.

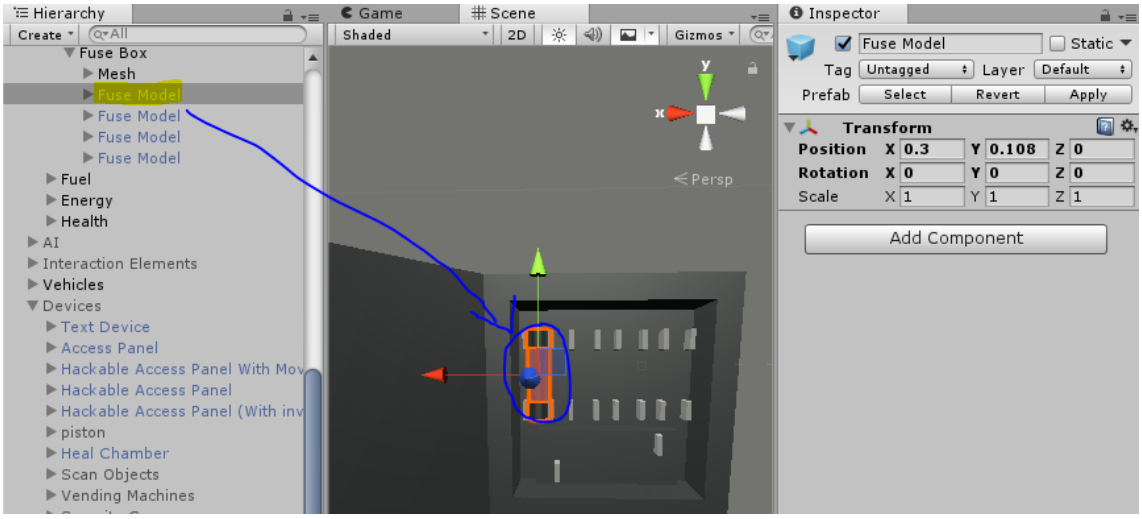


- Open the new element and, for this case, set to true the option Enable Object.

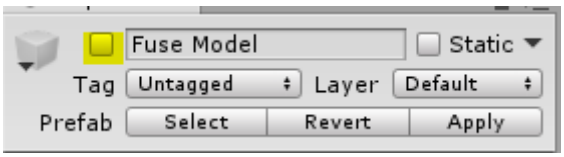


- This option activates a gameObject in the level and in this case, the fuses models placed in the fuse box are disabled, so when the player use a fuse from the inventory, the fuse is activated inside the box. For this example, the gameObject Fuse Model is disabled in the hierarchy. This is the element configured in the Object To Enable Field.



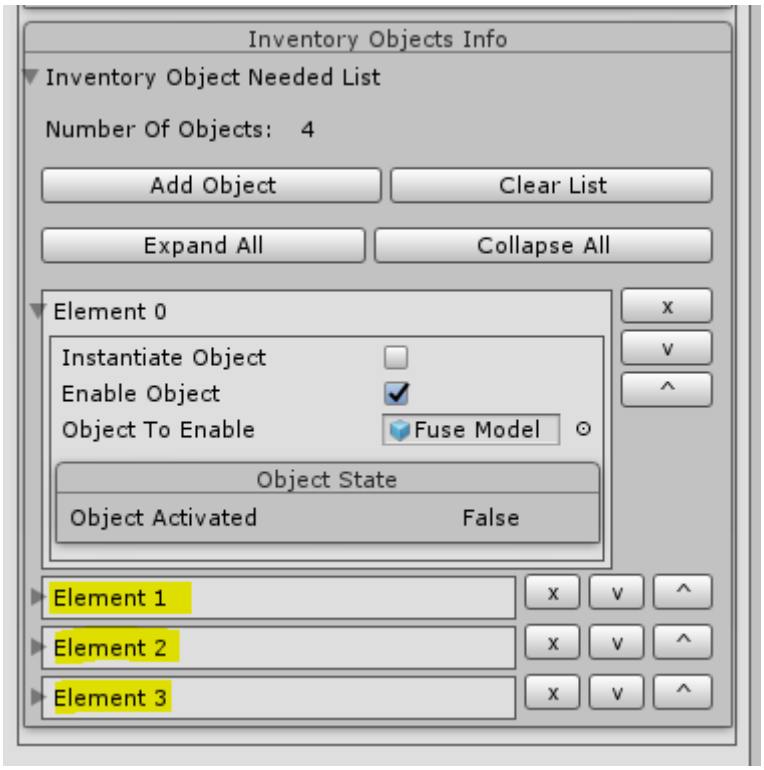


- The fuse model needs to be disabled before start the game.

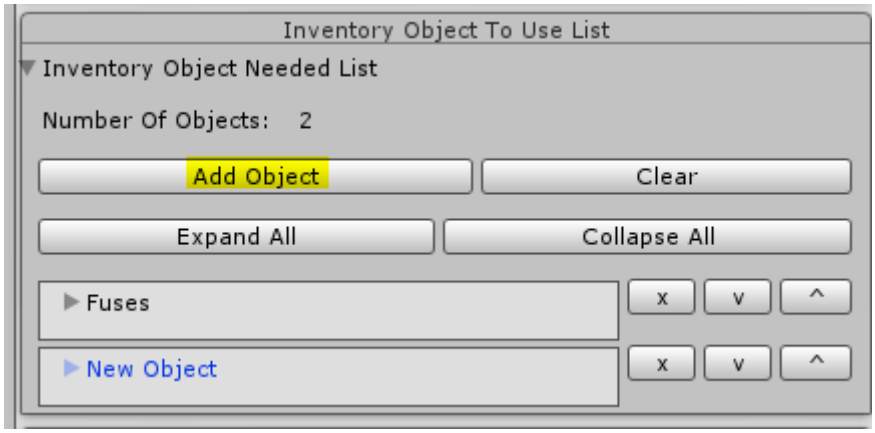


- Instead of activate disable gameObjects, there is an option to instantiate objects in the scene, but this will be explained later.

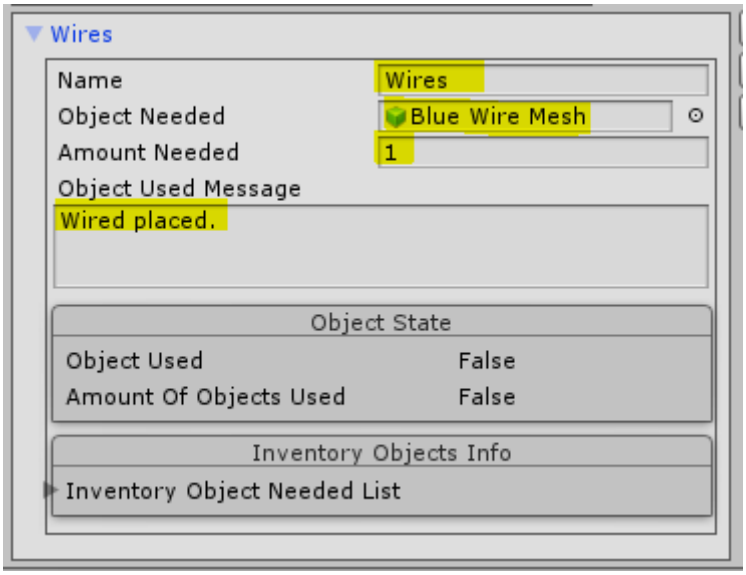
- In this case, there are four fuses, so configure the rest of them, like the first one.



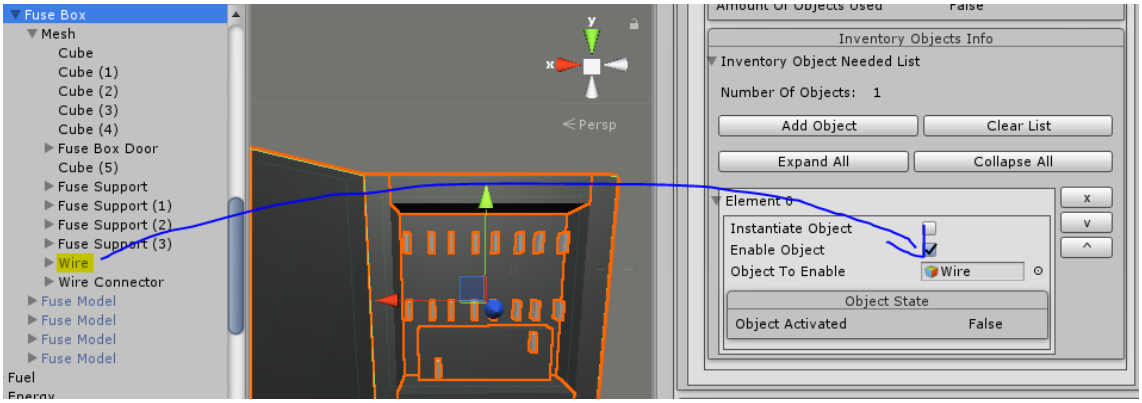
- Now, along with the four fuses, the fuse box needs a blue wire to work, so another object needs to be configured, but it is another element different from the fuses. Go to the main Inventory Object Needed List and press the Add Object button to configure the blue wire.

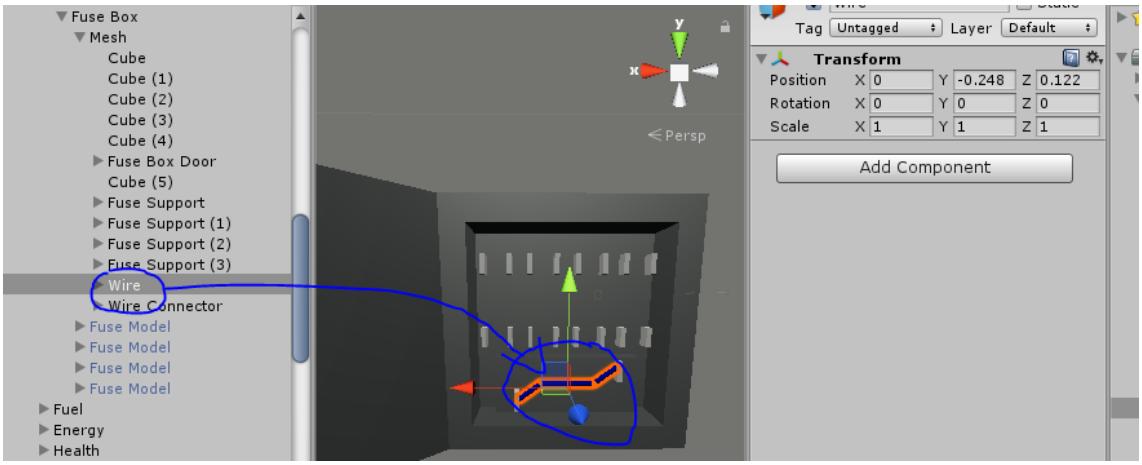


- For the blue wire, like for the fuses, configure the name, the message to show in screen and the amount needed and the inventory object mesh to use.

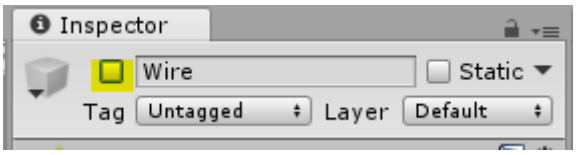


- And inside the Inventory Object Needed List, configure the object to active, in this case, a mesh of the wire already placed in the box, like the previous fuses objects.

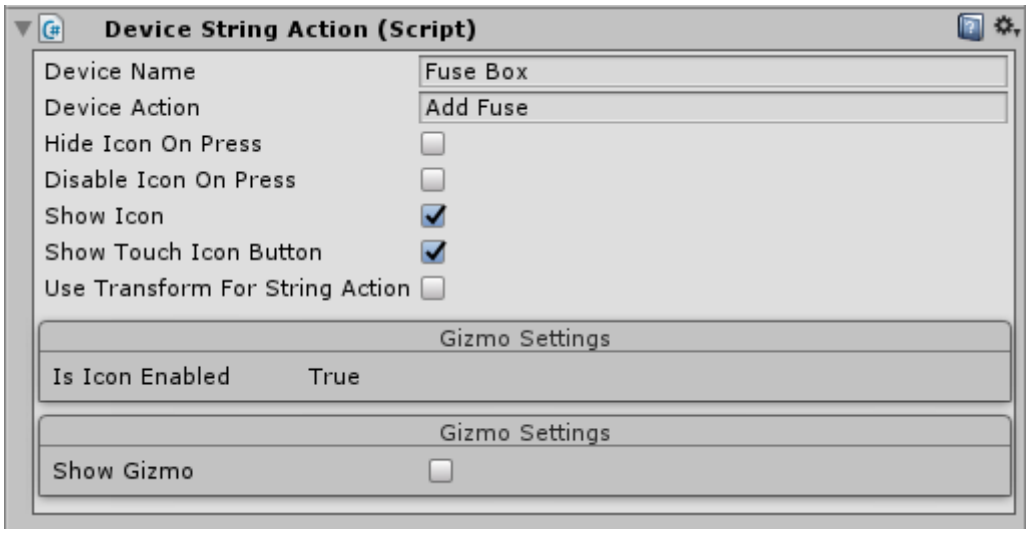




- Like with the Fuses Models configured before, the wire object needs to be disabled.



- Finally, add a Device String Action component to the parent, with the next configuration.

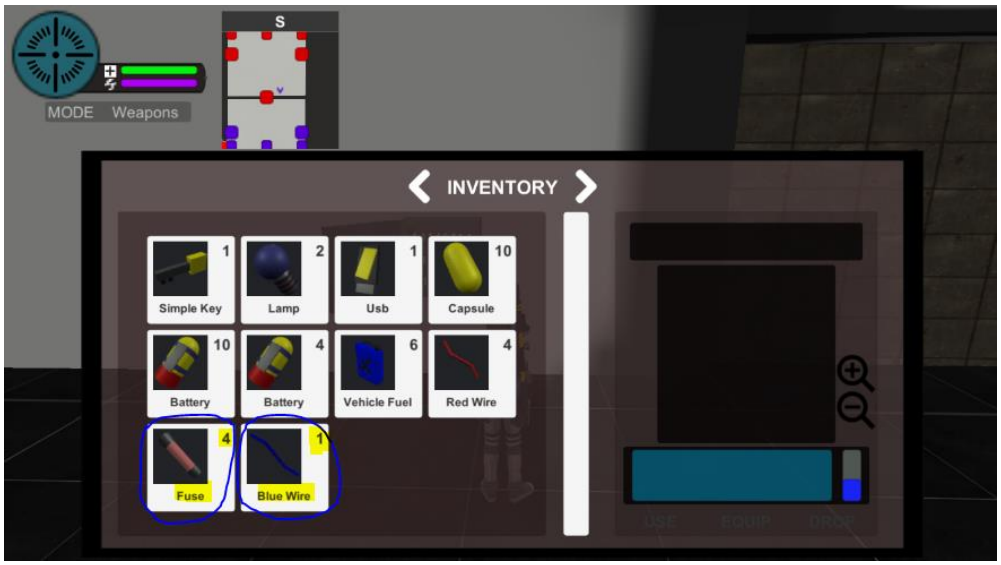


- The Device Name and the Device Action are the text shown in the icon of the interaction button on screen. The Show Icon is the default option to show that icon on screen. The Show Touch Icon Button is a default option used to show the touch button on mobile devices to use the interaction button.

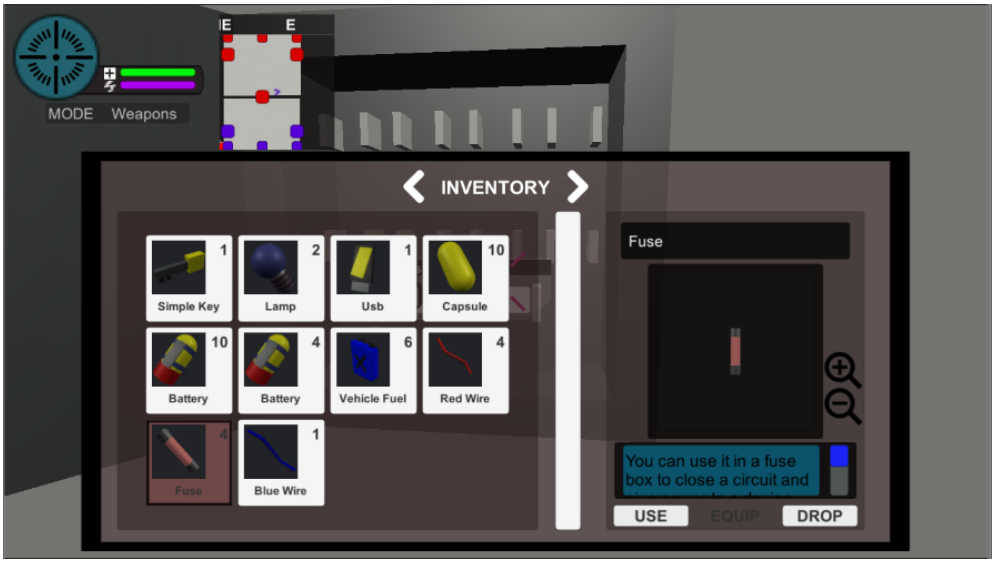
Check the Use Inventory Object works correctly

Once everything is configured, the fuse box can be tested. For this, go where it is placed and follow these steps:

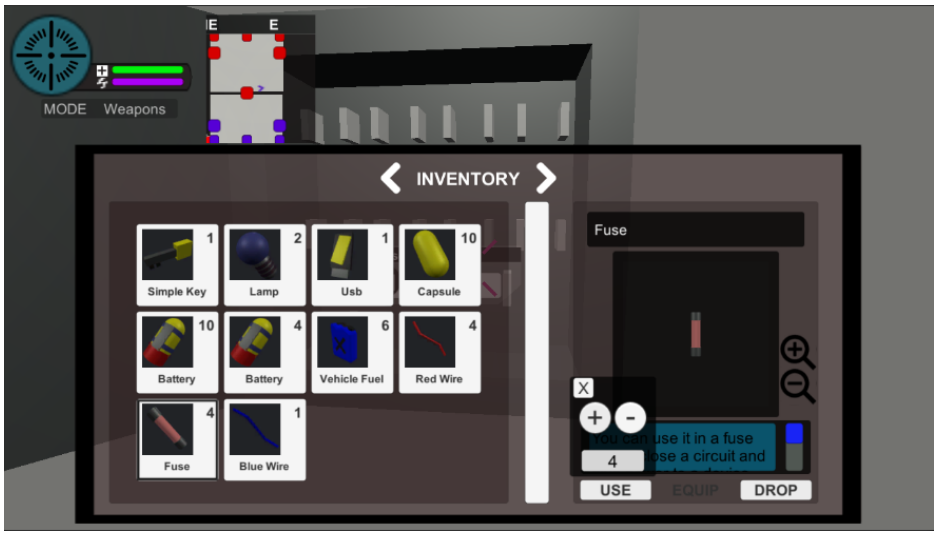
- Make sure to have at least 4 fuses and 1 blue wire in the inventory.



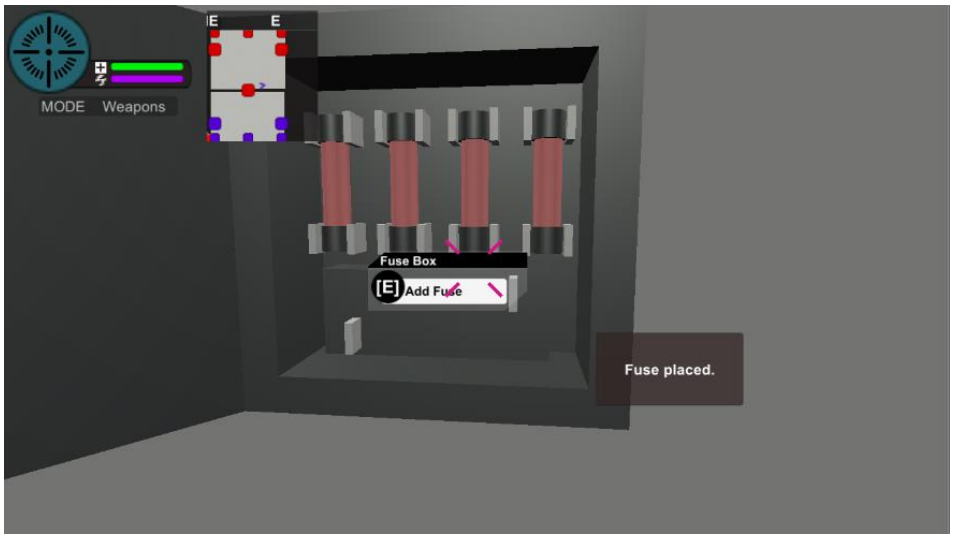
- Enter inside the device trigger, open the menu and press the Fuse object.



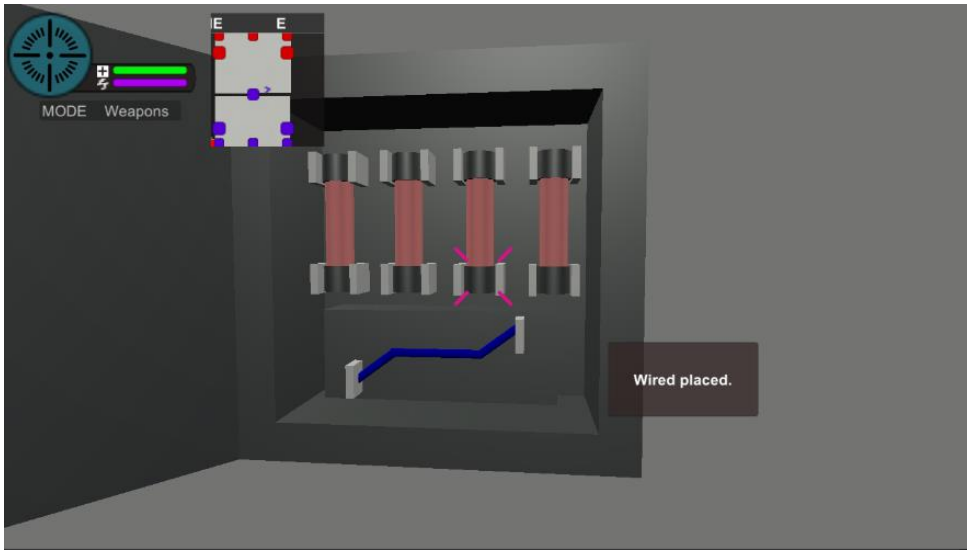
- Press the button Use, select the amount to use with the buttons + and - (it can be one by one, two, any amount from the available in the inventory). You can press also X to close the amount window.



- Press Use again. The fuses are activated and the message configure is shown.



- Open the inventory again, select the blue wire and use it.



As you can see, every message is shown according to the object used. Like this, all the element needed are placed and the door is unlocked. Also, in this case, when the door is unlocked, the map icon in the minimap changes.

WEAPONS

Warning: Some of these process will be automatized in the upcoming update 2.4b.

The player weapons are composed of these elements:

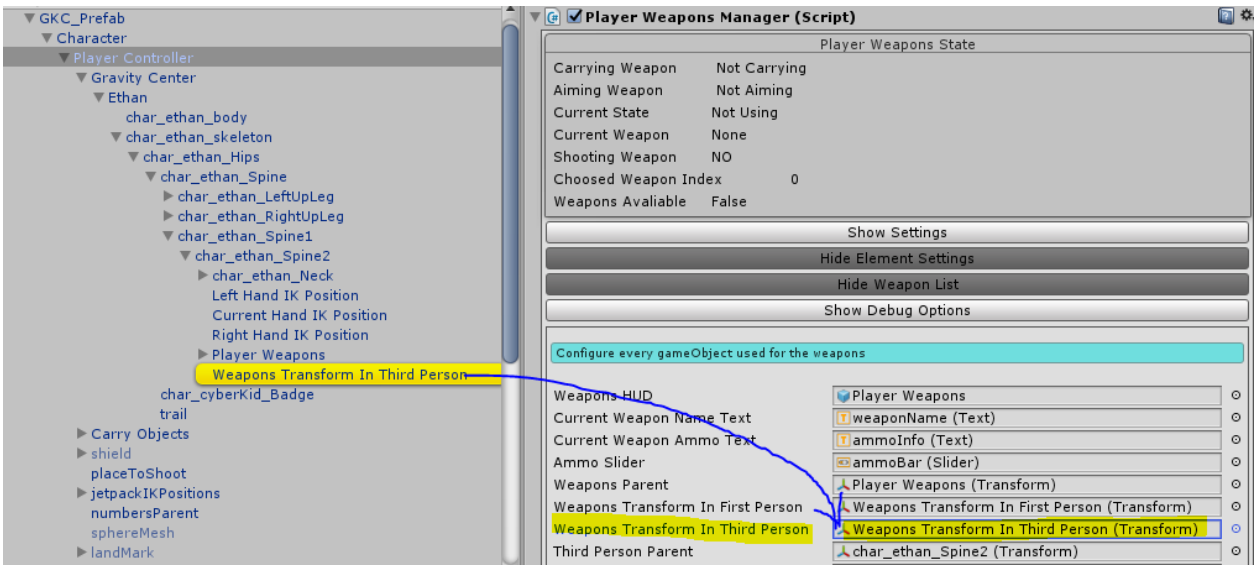
- **IKWeapon system**, it manages the IK configuration of the weapon, the movements to draw, aim, keep and do the recoil and other important elements to check the weapon state. It contains configuration for the waypoints used to move the weapons and the positions where the weapons is place when is drawn, when player walks, aims, shoot, etc…. It allows to configure the weapon in first and third person.
- **Player Weapons Manager**, it manages the whole weapon system, getting the input done by the player and applying it to the weapons, like shoot, change weapons, draw, keep, aim, drop, pick, etc… Also, it updates the HUD info regarding to weapons.
- **Player Weapon System**, it manages the weapon itself, containing the configuration of how the weapons fires, like fire rate, projectile to instantiate, all the physics made by the projectile, spread, scorches, sounds, auto shoot, etc….
- **Projectiles System**, it is the component used by any projectile in the asset. When a weapon is fired, the information configured about the type of projectile is sent to the projectile itself, like if the weapon fire projectiles by raycast or by applying speed, velocity, explosive settings, etc… and apply damages according to the type of surface detected.

Create new weapon

To create a new weapon, follow these steps:

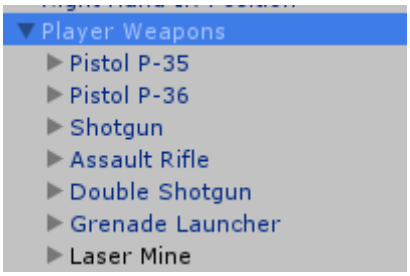
- Go to the weapons parent located in the skeleton of the player. This parent is called Player Weapons. It is located inside the spine, usually the second spine which is the chest of the model.

- You can find this element in the same parent that the gameObject called Weapons Transform In Third Person, used as a reference to know where the weapons are located in third person. This is configured in the Player Weapons Manager inspector in the Player Controller gameObject.



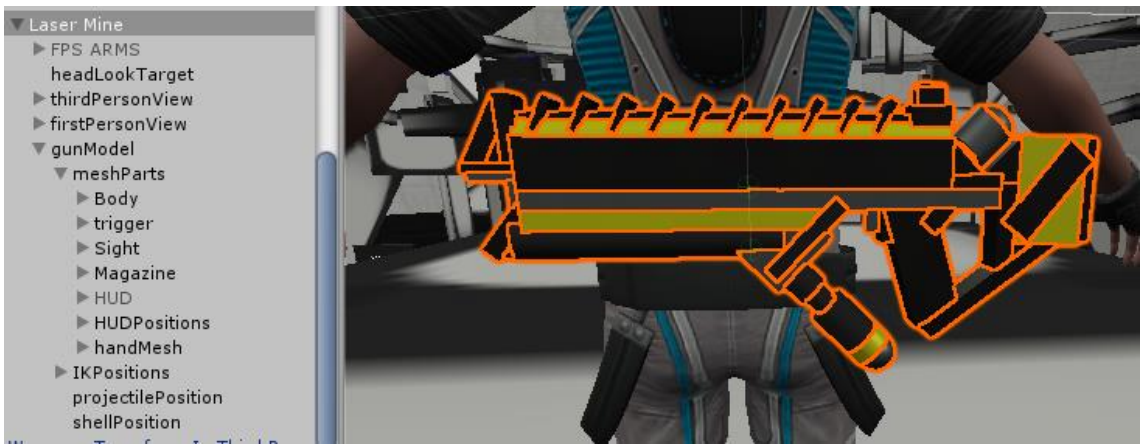
- Open the Player Weapons, which contains all the weapons that the player uses. Copy and paste in the same parent any of these weapons. If you are going to make a rifle type, you can copy Shotgun, Assault Rifle, Double Shotgun or the Grenade Launcher, so the

configuration of the player hands model in first person would be easier. In this case, let's copy the Assault Rifle and call it Laser Mine.

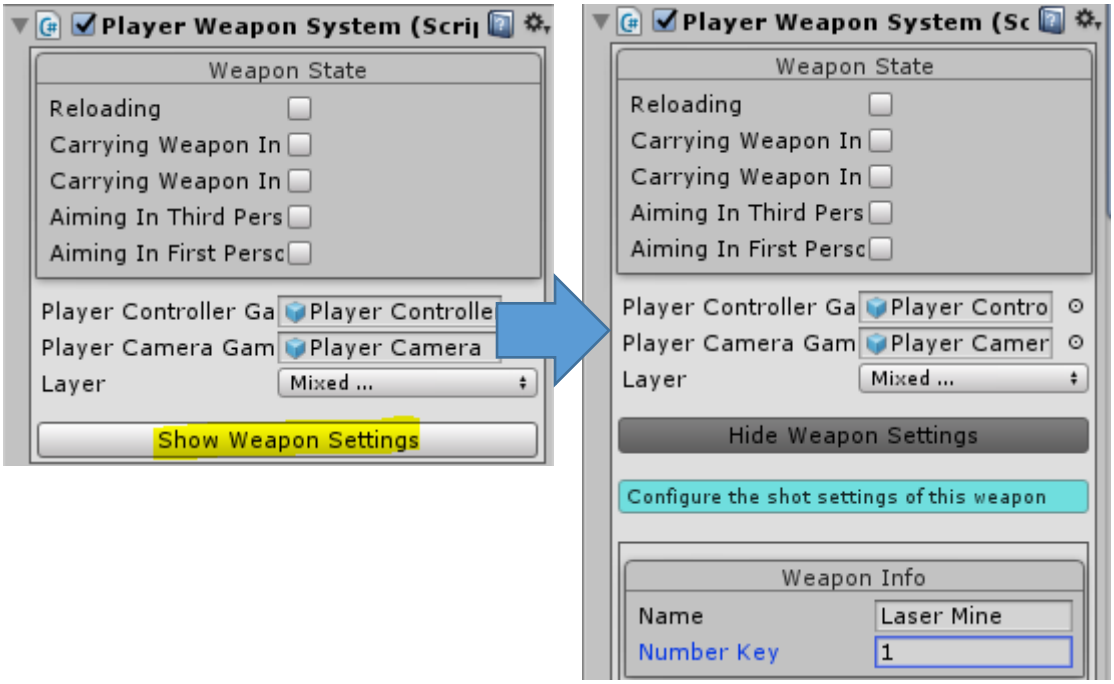


- To manage better the configuration of this weapons, disable the rest to have a better look (you don't need to do this if you don't want to).
- Now, open the Laser Mine gameObject and inside gunModel, there is the meshParts element, which contains all the meshes used in the weapon. The elements that can be removed are the next:

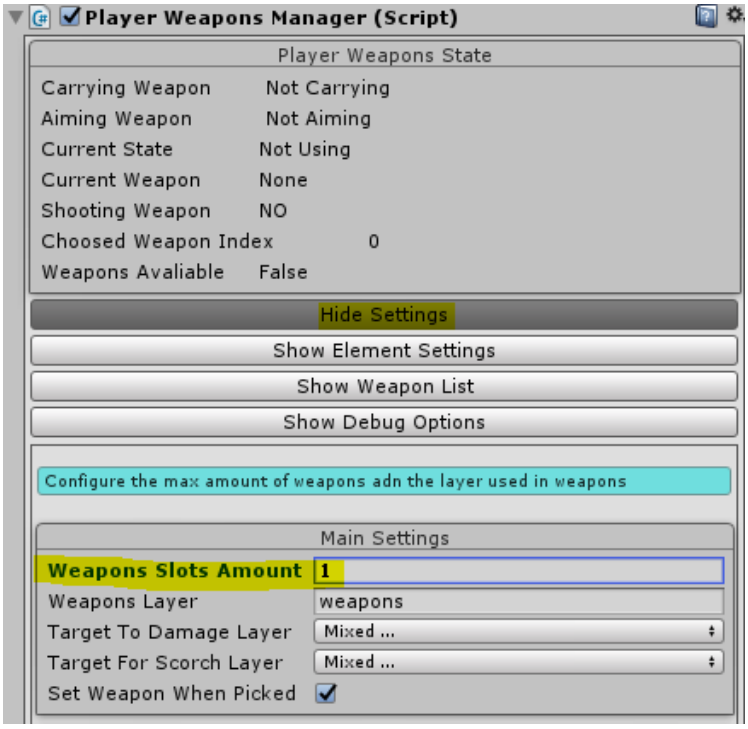
- Now, you can put your own weapon model inside meshParts gameObject. It is not necessary to separate the meshes into magazine, stock, etc…, it is just for a better organization of the elements. You can put a single mesh or maybe to, like the handle and the slide. In this case, this is the new weapon model added, separated into four parts (body, trigger, magazine and sight):



- Go the Player Weapon System inspector, in the gunModel gameObject, and configure the name of the weapon and the number key to use it, for example, the key 1, since this could be the first weapon. Inside the inspector, press the button Show Weapon Settings.

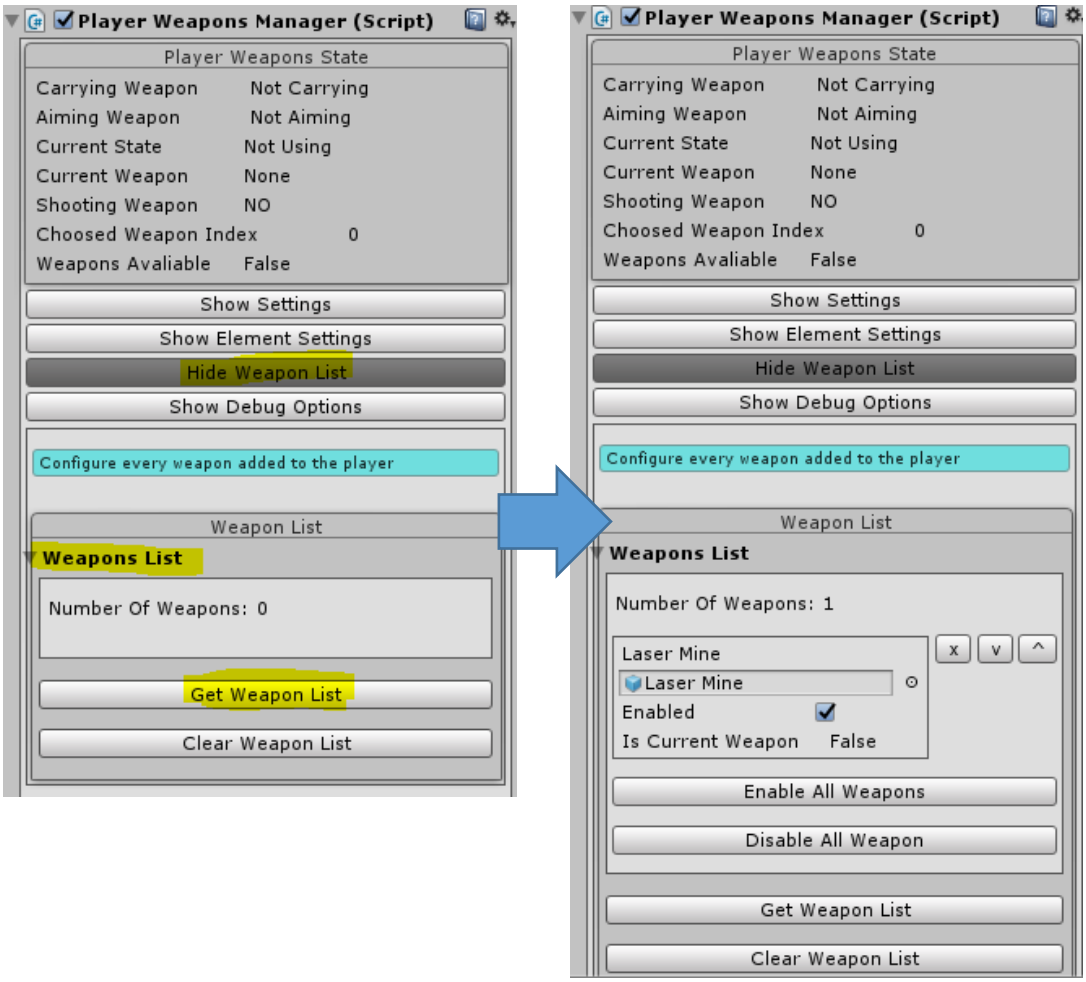


- Of course, the weapon number key can be 7 if you use the previous six weapons, or the next number of the amount of weapons that the player has in that moment.
- Once you configure all the weapons needed and they have been added to the Player Weapons Manager list, in that inspector, press the button Show Settings and set the field Weapons Slots Amount with the current number of weapons used by the player.



Add new weapon to Player Weapons Manager

- Go to Player Weapons Manager, Show Weapon List and press the button Get Weapon List to add this new weapon. You can have as many weapons as you need in the player, this process will add all of them to the manager.



- The player can already use this weapon like any other.

Configure third person settings

Every weapon can has a different shape and size, and the hands, arms and upper body of the player could need to be adjusted for a new weapon. For this adjustment, the best workflow, is to configure these elements in play mode, while the player is using the weapon to configure, copy the values in the Transform inspector, stop the game and paste values. For this, follow these steps:

Adjust IK positions

- Go to IKPositions inside the gunModel and adjust the hank and elbow IK positions according to the weapon size and shape. The empty transforms inside that parent are used as a position and rotation to be followed by hands and elbows.

- For example in the image below, the left hand is in a bad position since it is crossing the weapon mesh, so the gameObject leftHand needs to be placed a little down.

- After move the leftHand transform, this is how it looks.

- The same would be done with the rest of IK positions until the player holds the weapon correctly.

This configuration can be done while the player is holding the weapon, or aiming, or shooting, these positions are the same for all the actions with that weapon in third person, so it doesn't matter how it is configured.

Adjust head look direction

Inside the weapon, there is a gameObject called headLookTarget, used as a reference point to make the player head look into it when he aims.

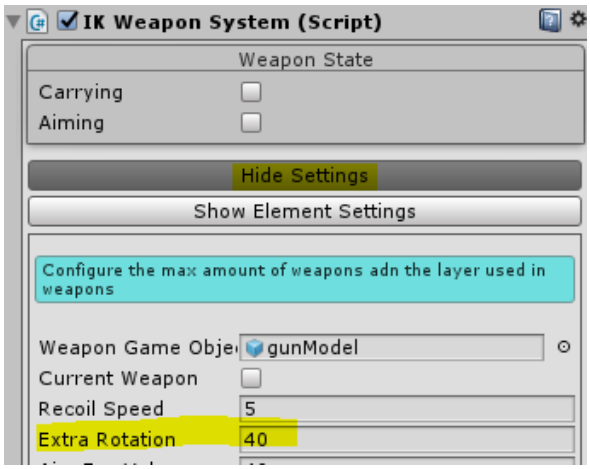
This point can be moved to make the head look higher or lower or to any direction, but usually doesn't need a further configuration.

Adjust upper body rotation

Weapons like the used in this tutorial is big, and needs to be used with two hands, so like in real life, when a person holds this kind of weapons, the upper body is rotated in the horizontal direction, moving a shoulder backward and the other forward. Without this, the player would look like this:

This rotation needs to be configured to get a natural position in the model. For this, follow these steps:

- In the gameObject that contains the whole weapon, in this case, Laser Mine in the IK Weapon System inspector.
- Press the button Show Settings and set the rotation amount in the player's body the field Extra Rotation.



- A good value is a rotation between 35 – 45 degrees. So the player looks like the images below.

For little weapons, this rotation maybe is not necessary, so the value configured is 0.

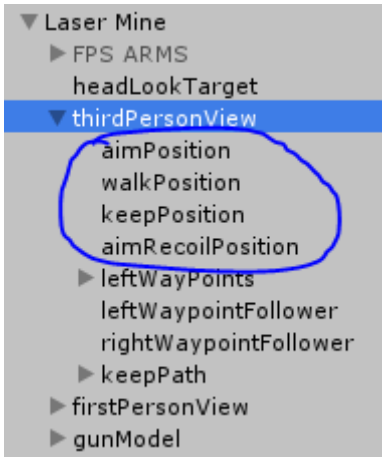
Adjust weapon positions in third person

It will be explained further soon in a new doc version.

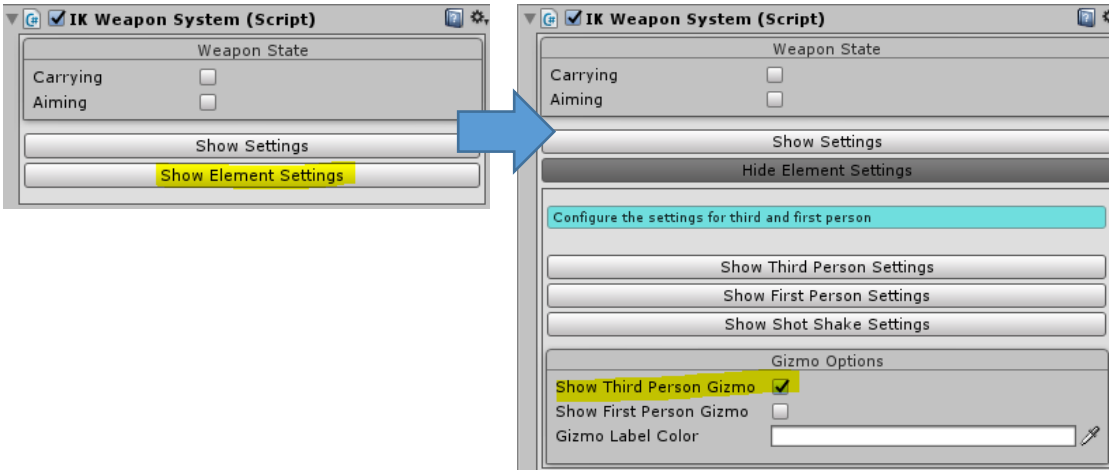
There are four positions and rotations configured for every weapon. These points are:

- Keep, where the weapon is stored while is not used.
- Walk, where the weapons is placed when the player is holding it while he moves or is in idle state.
- Aim, where the weapon is place when the player aims.
- Recoil, where the weapon is placed when is fired.

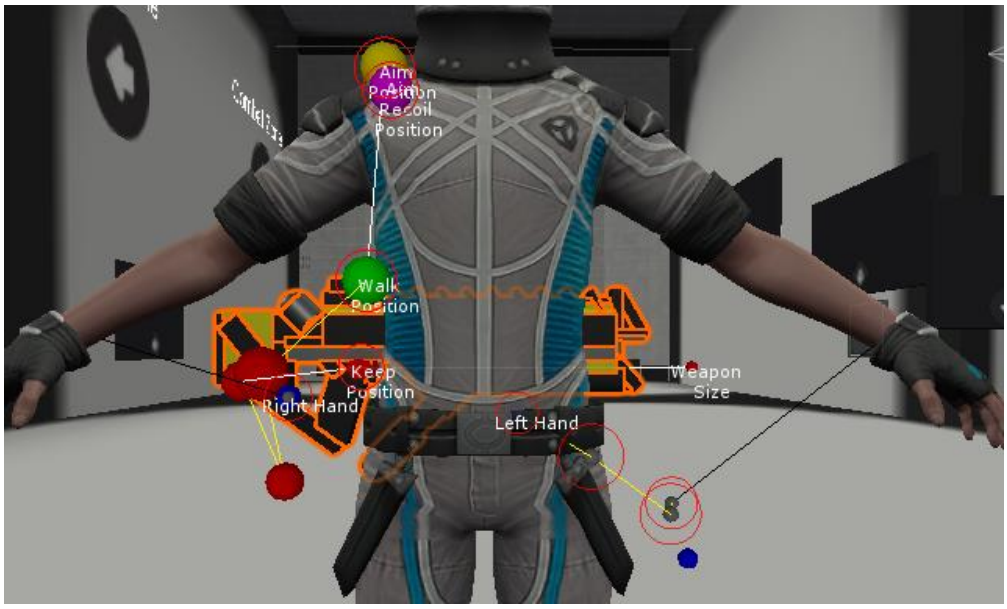
These points are placed inside the weapon, in the gameObject thirdPersonView.



To configure them, go the weapon, and in the component IK Weapon System, press the button Show Element Settings and then active the third person gizmo.



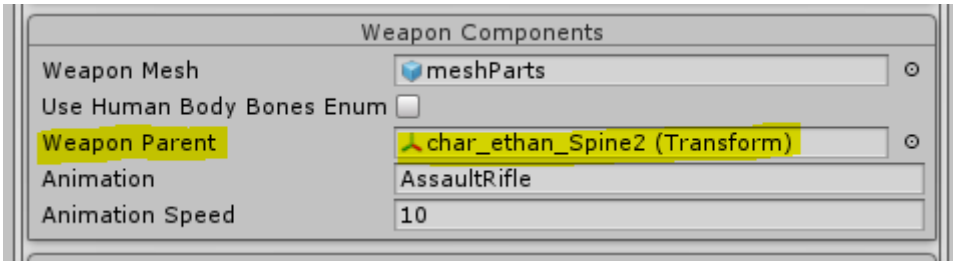
This gizmo helps to see better where every point is placed for an easier configuration.



The configuration needed here is to move and rotate these four points as you need according to how the player use and move these weapons. The weapon will move from point to point including position and rotation.

Configure weapon position in player's body

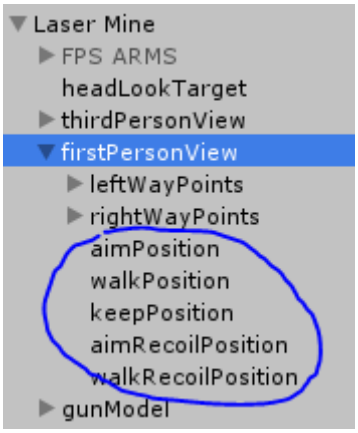
According to the size of the weapon, it is usually placed in an upper leg or the back of the player, so when he moves, the weapons follow these movements. To attach the weapon to a desired body part, go to the gameObject gunModel, to the inspector Player Weapons System, Settings and the part Weapons Components.



Select the bone to place the weapon and it will be attached there when the games starts.

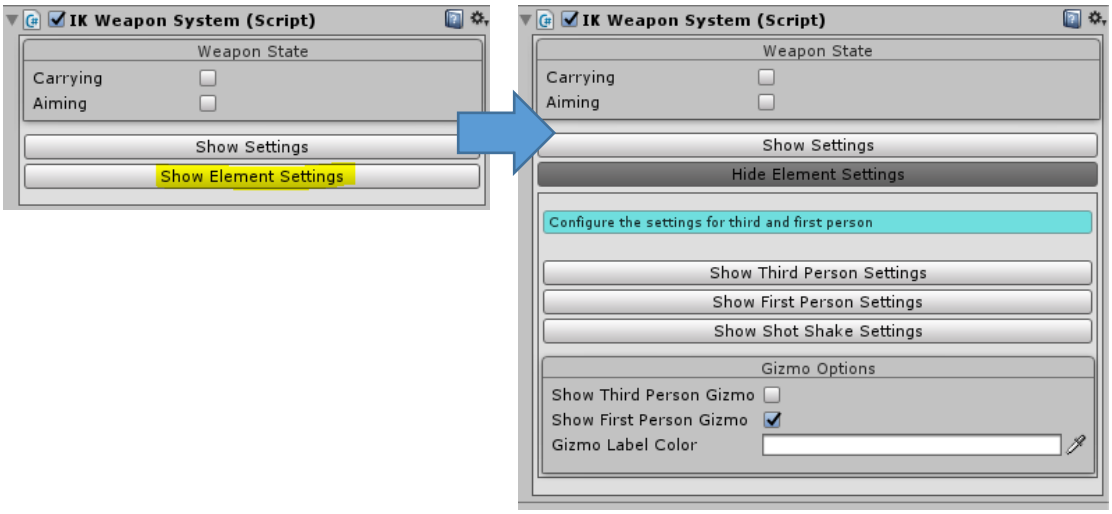
Configure first person settings

In first person, there is the same group of positions to configure where to place the weapon according to the action in that moment. These positions are inside the weapon, in the gameObject firstPersonView.

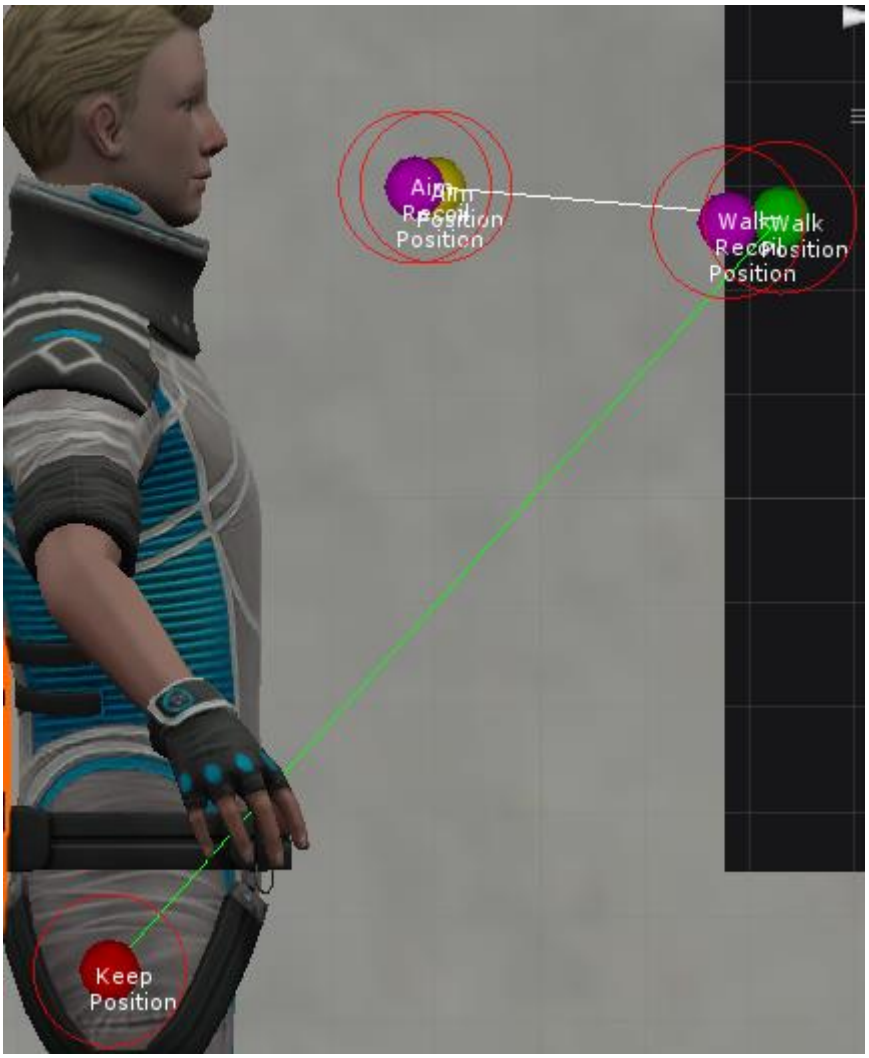


There is a new position, walkRecoilPosition, since in this view, the player can shoot if he is aiming or just holding the weapon.

To configure them, go the weapon, and in the component IK Weapon System, press the button Show Element Settings and then active the first person gizmo, in the same part that third person (disable the third person gizmo for a better view).



In this case, the view is simpler.



So like in third person, here you only need to move and rotate these points according to how the weapon needs to be placed in first person view.

Enable or disable a weapon at the beginning of the game

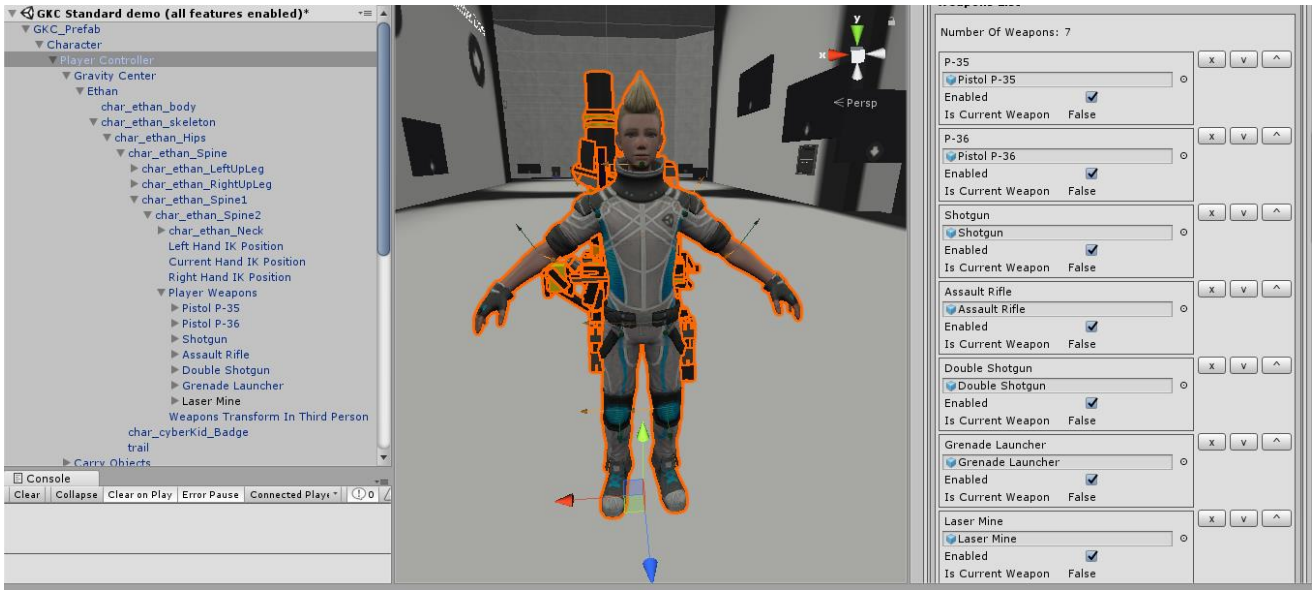
In the Player Weapons Manager, every weapon added in the list has a field to configure if that weapon is enabled when the game start, so the model of the weapon is visible in the player's body and he can use it.

If the weapon is disabled, but it is in the list of weapons, it can be enabled and used if the player picks that weapon in the scene.

If the weapon is placed in the weapons parent but it is not configured in the list inspector, it won't be used by the player even if that weapon is a pickup in the scene.

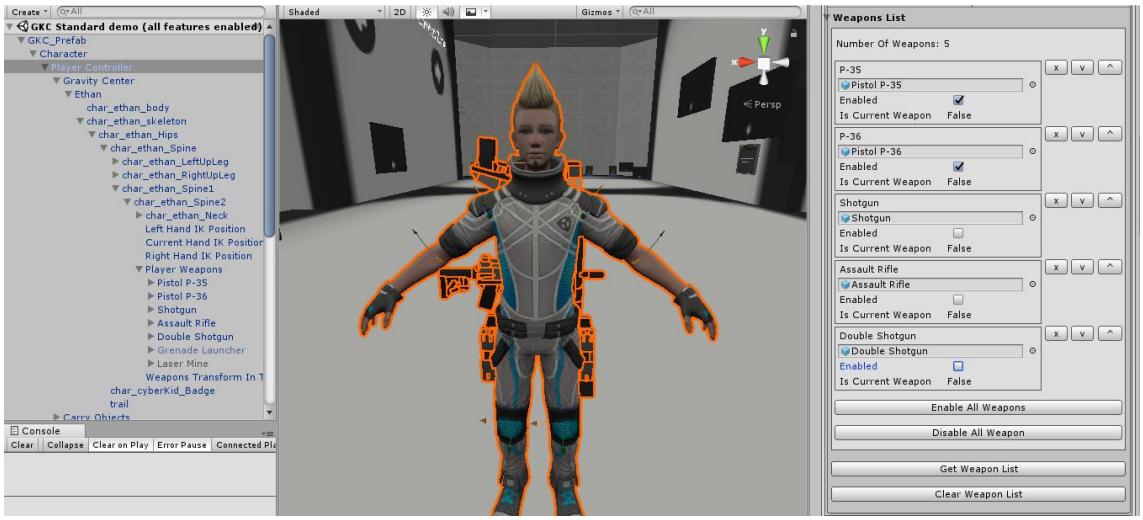
Before the game starts, all the weapons gameObject needs to be enabled, even if in the list inspector is configured as disabled to be picked later.

In this example, the player has 7 weapons in the list all of them enabled, so they are available from the beginning of the game (notice that all weapon gameObjects are enabled).



So when the game starts, this is what it looks.

In this other example, the player has 5 weapons and the last 3 are disabled in the list.



And this is what happens when the game starts:

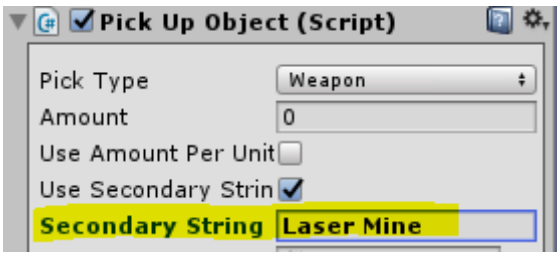
As you can see, only the two pistols are enabled. The other 3 can be activated if the player picks them in the level.

Make the prefab of a new weapon

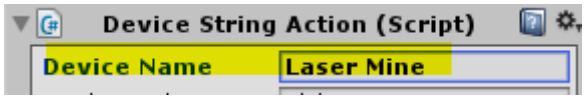
After a new weapon is created and configured, you need to create its prefab if you are going to use that weapon as a pickup or to make the player able to drop that weapon.

For this, follow these steps:

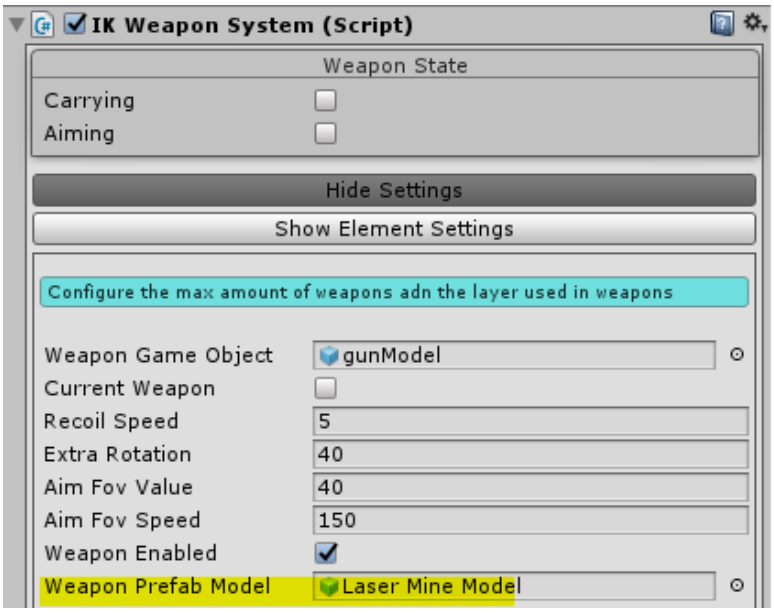
- Go to the folder Game Kit Controller/ Prefabs/ Player Weapons and grab and drop any of them.
- Remove the meshParts gameObject.
- Copy and paste the meshParts of the new weapon inside that prefab.
- Make sure the local position and rotation of the meshParts place in the prefab is (0,0,0)
- Adjust the box collider according to the size and shape of the new weapon.
- In the Pickup Object inspector of the prefab, configure the name of the weapon in the field Secondary String.



- Go to the child trigger inside the prefab and in the inspector Device String Action, configure the name of the weapon in the field Device Name.



- Grab and drop the prefab in the same folder of the other weapons.
- Finally, assign that prefab in the IK Weapon System of the new weapon, in the field Weapon Prefab Model.

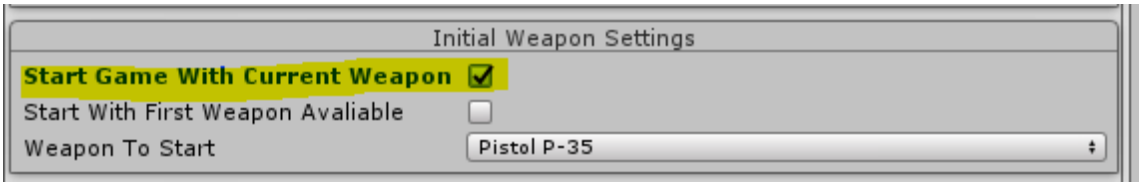


Like this, if the player drops the weapon, the object instantiated is a copy of this prefab and the weapon in the player's body is disabled until the weapon is picked again.

This same prefab can be used to be dropped by enemies when they died, place it inside a chest, a crate, etc….

Start game with a certain weapon

To start the game with the player carrying a weapon, go to Player Weapons Manager, press the button Show Settings and in the part of Initial Weapon Settings, activated the field Start Game With Current Weapon.



Then select if the player will use the first weapon enabled in the list manager or select the weapon to start from the list of weapons added in the manager.

The player will start with that weapon in any view, no matters if the game starts in third or first person.

Warning: next and more elements of the weapons will be explained in a newer version of this document.

Enable and disable weapons without remove them

Configure draw/keep position path for weapons in third person

Configure weapon sway

Configure first person arms

Configure weapon shake camera

Configure weapon projectiles

Add new ability to projectiles

POWERS

- POWER LIST
- POWER MANAGER LIST

COMBAT

FOOT STEPS

DECALS MANAGER

SAVE SYSTEM

EVENT TRIGGER SYSTEM

SOUNDS EFFECTS

Some of the sounds effects in the asset have been obtained from a free sounds effects page: http://www.freesfx.co.uk.

SUGGESTIONS, ISSUES OR PROBLEMS

Feel free to post any comment in the asset forum.

Asset forum: https://forum.unity.com/threads/game-kit-controler-1st-3rd-controller-with-weapons-vehicles-2-3-8-released-features-poll.351456/

Also, you can send me a PM, my username is sr388.

If you have any doubt problem or suggestions, please send me an email: santimonti90@gmail.com

SOCIAL MEDIA

Asset YouTube Channel: Two Cubes Studio

https://www.youtube.com/channel/UCs21at6NKl_ieSIbE_2unuA

In this YouTube channel will be upload different tutorial videos for every system of this asset, so check it out.

Also, in Twitter and Instagram I post gifs and videos with progress of the asset.

Twitter: @santimonti90

https://twitter.com/santimonti90

Instagram: @santimonti388

https://www.instagram.com/santimonti388