Third Person Controller – Shooter Template

(v2.5.1 - 01/05/2020)

Thank you for supporting this asset, we developed this template because a lot of developers have good ideas for a Third Person Game, but building a Controller is really hard and takes too much time.

The goal of this template is to deliver a top quality controller that can help those who want to make a Third Person Game but are stuck trying to make a controller.

With this template, you can set up a 3D Model in just a few seconds, without the need of knowing advanced scripting or wasting time dragging and dropping game objects to the inspector, instead you can just focus on making your game.

--- Invector Team ---

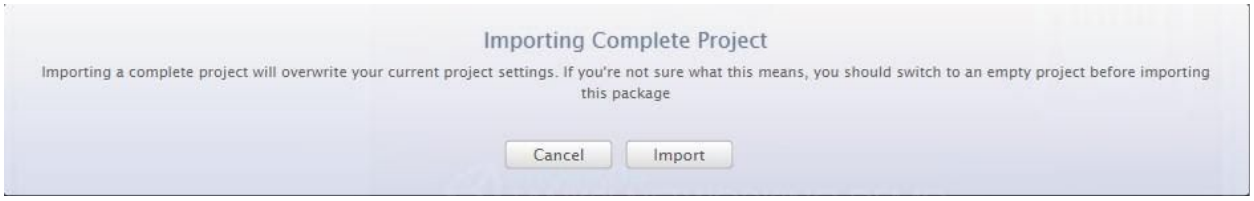
Ps This Documentation will cover the **Shooter Features**, there is another documentation for the **Basic Locomotion** and **Melee Combat** on their respective folders.*

Summary

FIRST RUN

IMPORTANT

This is a **Complete Project**, and as every complete project it includes a custom **InputManager**, **Tags**, **Layers**, etc… **Make** sure that you import **on** a **Clean Project**.



Importing Complete Project

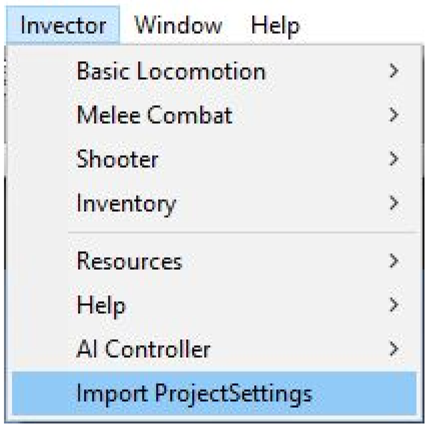
Importing a complete project will overwrite your current project settings. If you're not sure what this means, you should switch to an empty project before importing this package

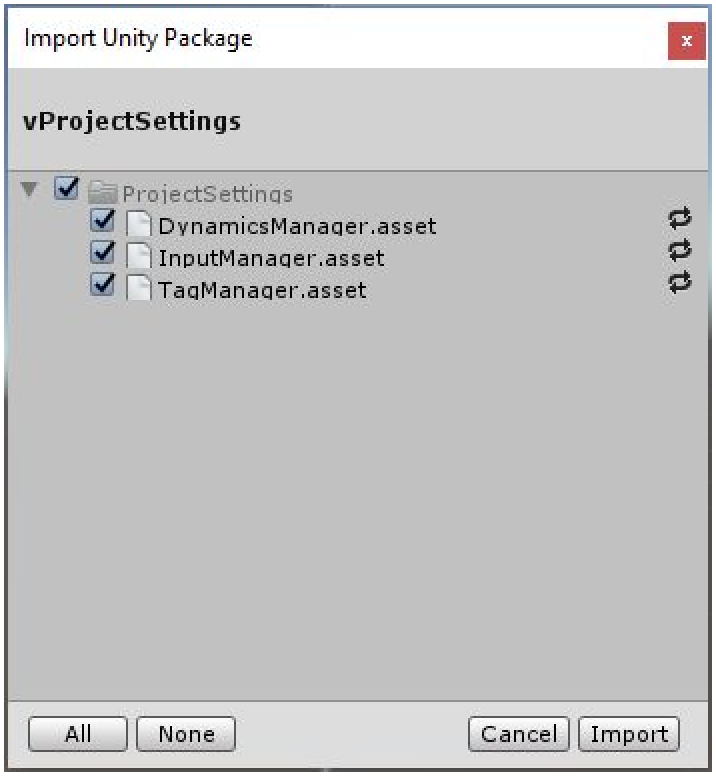
- *Importing on an existent project*

There are basically 3 files that are **extremely necessary for the correct functioning of this template**.

- ***DynamicsManager****.asset* - this will apply correct all the Collision Matrix of our Layers, for example we need the layer "Triggers" to not collide with the layer "Player".
- ***InputManager****.asset* - We have a custom input mapped with the Xbox360 controller, if you don't input those 2 files, the template will present errors and undesired behaviour.
- ***TagManager****.asset* - Includes all the necessary Tags and Layers for the project to work correctly.

After importing the template you can manually import those files by going to the tab **Invector > Import ProjectSettings.**




Import Unity Package

vProjectSettings

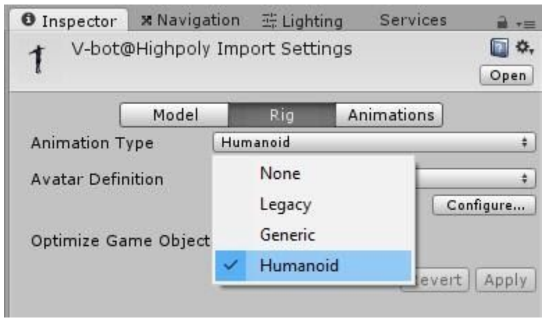
ProjectSettings
- [✓] DynamicsManager.asset
- [✓] InputManager.asset
- [✓] TagManager.asset

Now that you have imported the necessary files, you can explore the several demo scenes and figure it out what kind of Third Person Game you want to create.

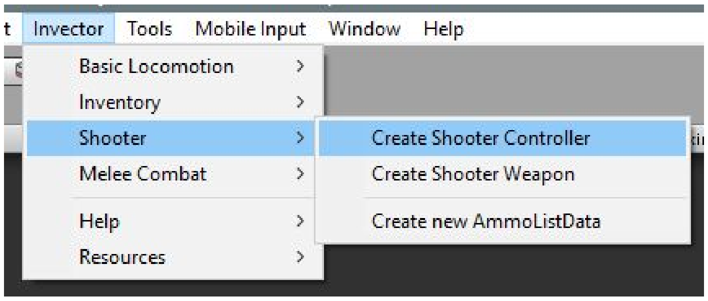
***Updates also need to be imported into a Clean Project, so MAKE SURE TO BACKUP your previous project and transfer the necessary files to your new project. ***

CREATING A NEW SHOOTER CONTROLLER

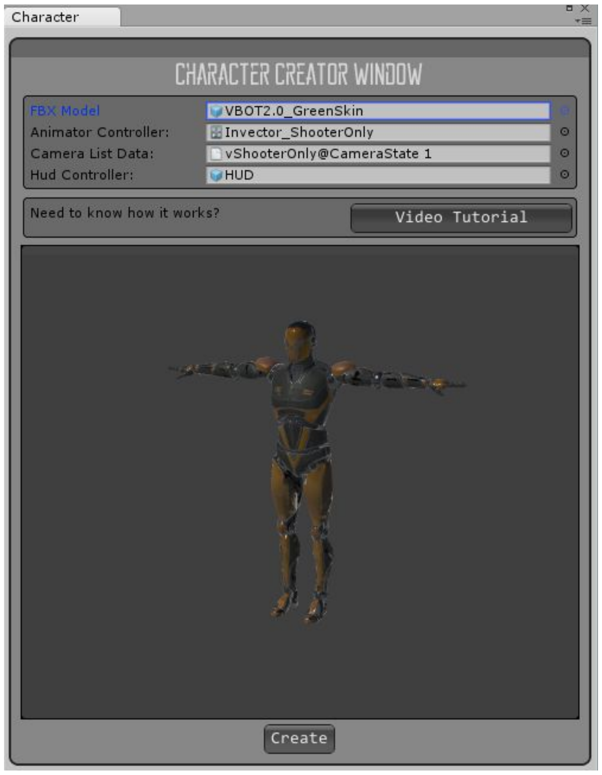
Make sure that your fbx character is set up as **Humanoid**



To setup a new character, go to the tab *Invector* > *Shooter* > *Create Shooter Controller*



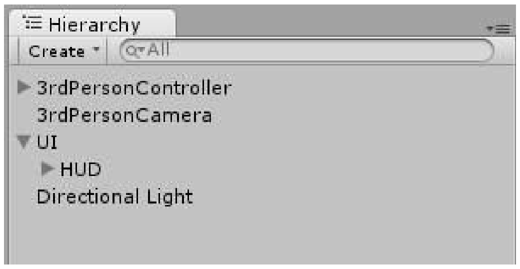
Make sure your Character is **Fully Rigged** and set up the FBX as a **Humanoid**, then assign the FBX to the field “FBX Model” and click on the button “Create”.



FBX Model: VBOT2.0_GreenSkin
Animator Controller: Invector_ShooterOnly
Camera List Data: vShooterOnly@CameraState 1
Hud Controller: HUD

Ps Make sure to select the **Invector_ShooterOnly or Invector_ShooterMelee** if you want to use both shooter and melee as your Animator Controller, you can find the file at the folder: Shooter > Animator, or just click on the little circle icon.*

The **Character Creator** window will take care of all the hard work automatically and set up components such as capsule collider, layers, tags, rigibody, etc... It will create the **ThirdPersonController**, **ThirdPersonCamera** and a UI Canvas with a **HUD** to display health, stamina and other information.



Your Capsule Collider settings will be based on your model proportions, if the capsule gets the wrong size, make sure that you rig is correct, and that your **model is using the correct Scale Factor** the same goes if the ragdoll **gets** weird.

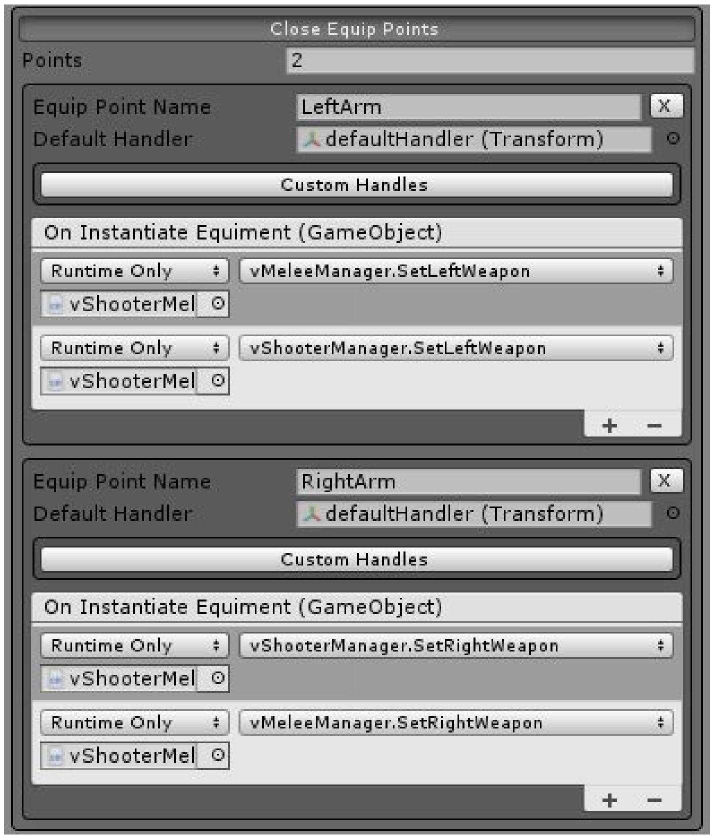
Hit Play and enjoy ☺

HOW TO ALIGN A SHOOTER WEAPON (RIGHT HAND)

After creating your character, you will need a *'handler'* to manage the position/rotation of the weapons you instantiate in game.

A **Handler** is basically an empty transform located inside your character's left and right hands, you will first align this handler into a position where a weapon looks correct in the characters hand and all weapons will be instantiated there when you pick up the collectible.

If you choose to use our **Item Manager**, both handlers will be automatically created for you once you attach the ItemManager to your character and you can see them or create customizable handlers for specific weapons in the "**Open Equip Points**" tab.


Close Equip Points
Custom Handles
On Instantiate Equiment (GameObject)
Custom Handles
On Instantiate Equiment (GameObject)

Points: 2

Equip Point Name: LeftArm
Default Handler: defaultHandler (Transform)

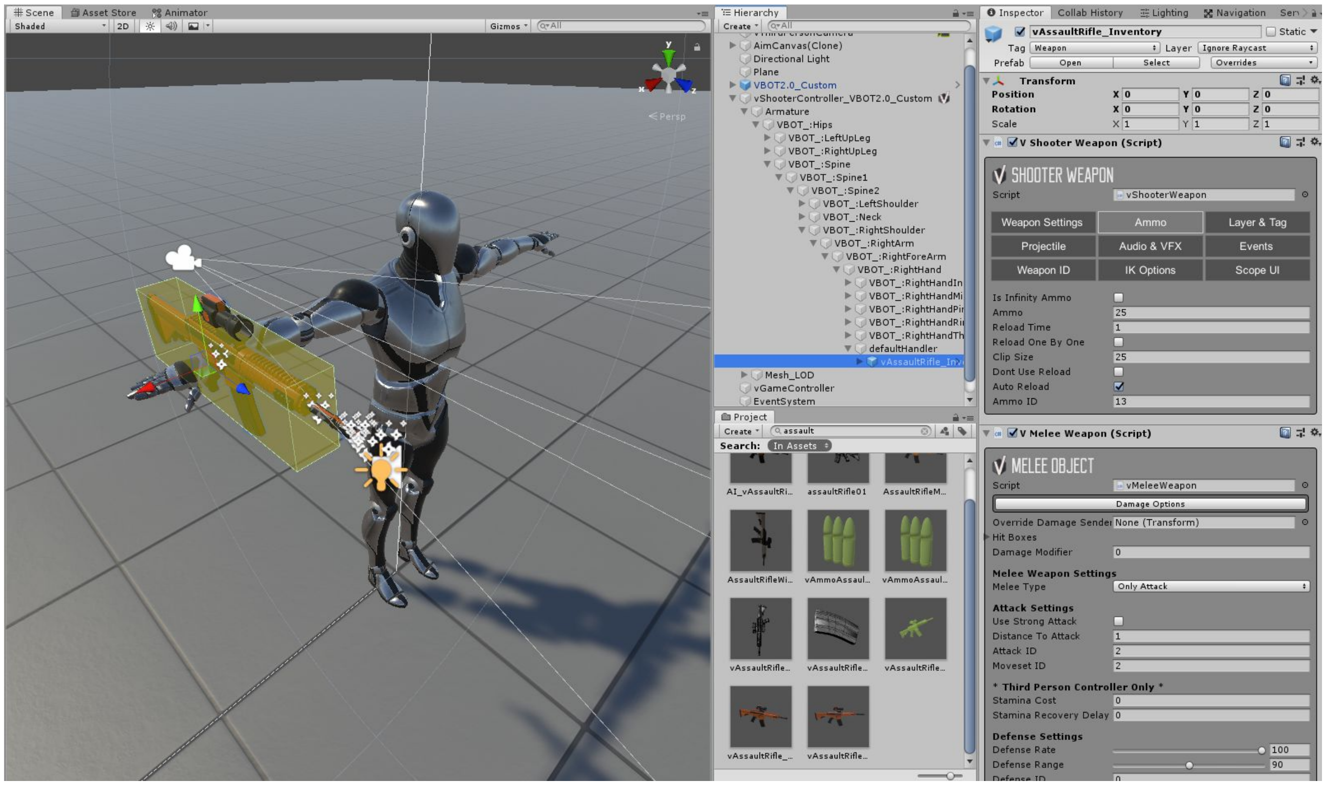
Runtime Only | vMeleeManager.SetLeftWeapon | vShooterMel
Runtime Only | vShooterManager.SetLeftWeapon | vShooterMel

Equip Point Name: RightArm
Default Handler: defaultHandler (Transform)

Runtime Only | vShooterManager.SetRightWeapon | vShooterMel
Runtime Only | vMeleeManager.SetRightWeapon | vShooterMel

Shooter Weapons will be equipped in the **RightArm defaultHandler**, so select this transform in the hierarchy and drag and drop any Weapon Prefab inside the defaultHandler.

For example search in the Project window to “**vAssaultRifle_Inventory**”, drag and drop inside the ‘**defaultHandler**” and reset it’s position and rotation back to 0, 0, 0.



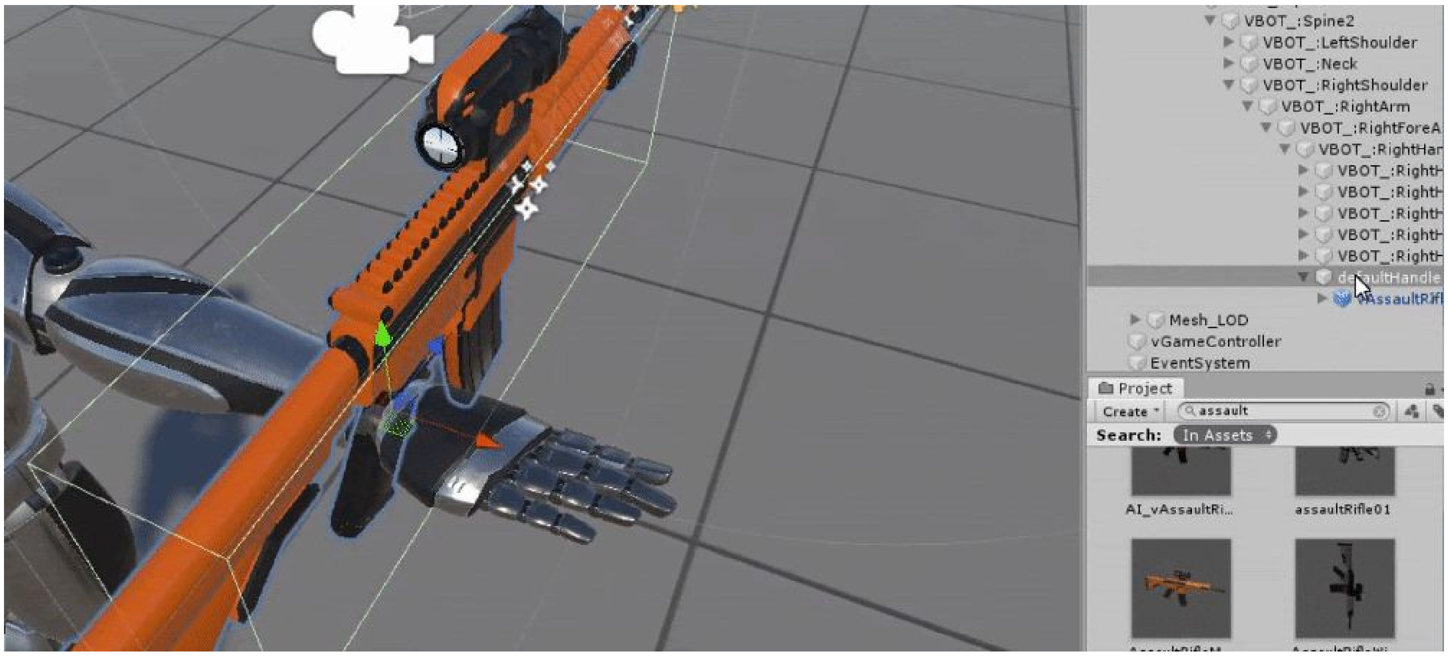
vAssaultRifle_Inventory
Tag: Weapon — Layer: Ignore Raycast

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

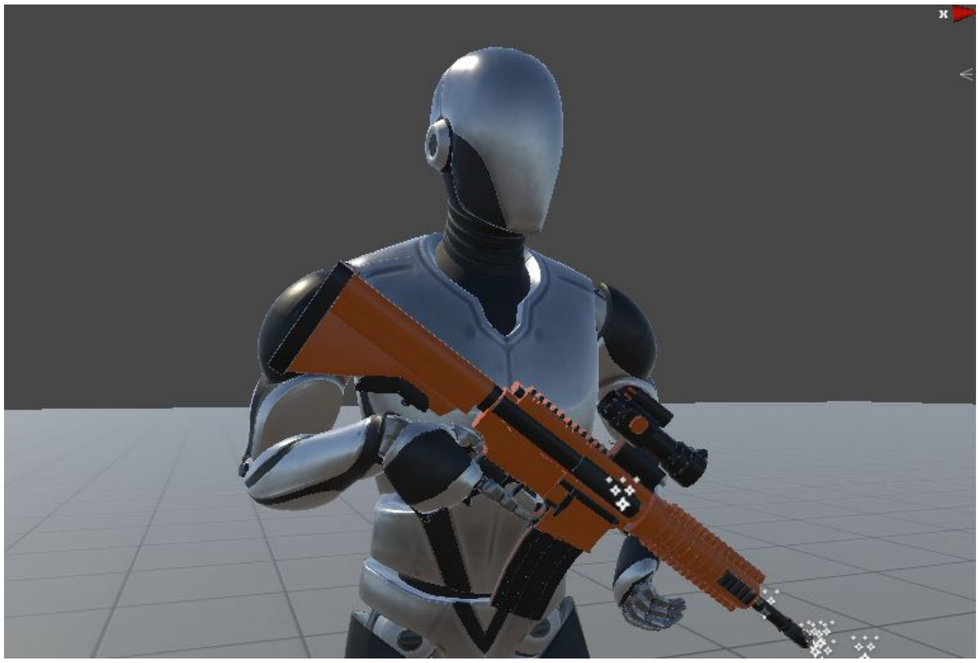
V Shooter Weapon (Script)
Script: vShooterWeapon
Is Infinity Ammo: ☐
Ammo: 25
Reload Time: 1
Reload One By One: ☐
Clip Size: 25
Dont Use Reload: ☐
Auto Reload: ✓
Ammo ID: 13

V Melee Weapon (Script)
Script: vMeleeWeapon
Override Damage Sender: None (Transform)
Damage Modifier: 0
Melee Weapon Settings
Melee Type: Only Attack
Attack Settings
Use Strong Attack: ☐
Distance To Attack: 1
Attack ID: 2
Moveset ID: 2
* Third Person Controller Only *
Stamina Cost: 0
Stamina Recovery Delay: 0
Defense Settings
Defense Rate: 100
Defense Range: 90
Defense ID: 0

Now you can manually align the ‘defaultHandler’ to a position close to where the weapon should be located.
* **ATTENTION**: You should **<u>NOT</u>** move the Weapon Prefab itself, this gameObject should be at 0,0,0, **you will only move/rotate the Handler**, this way all the weapon prefabs that will be instantiated inside the handler will be aligned as well.

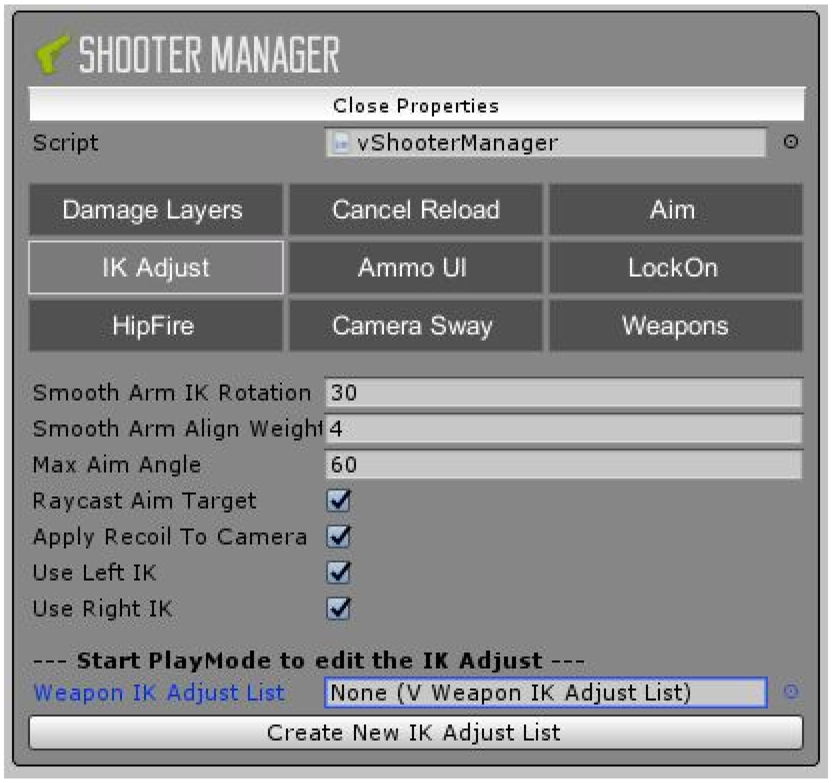


You can now hit **Play** to see if the **weapon position in the RightHand looks good**, it doesn't need to look perfect because we're still going to the **IK Adjustments** where we will **align the LeftHandIK.**



HOW TO ALIGN THE LEFT HAND IK FOR ALL WEAPONS

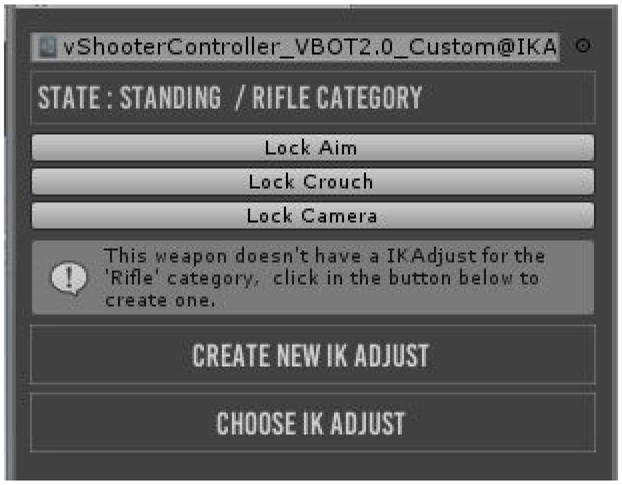
Go to your ShooterManager inspector and select the tab “IK Adjustment” and hit the button to create a new “IK Adjust List”.



Script: vShooterManager
Smooth Arm IK Rotation: 30
Smooth Arm Align Weight: 4
Max Aim Angle: 60
Raycast Aim Target: ✓
Apply Recoil To Camera: ✓
Use Left IK: ✓
Use Right IK: ✓
--- **Start PlayMode to edit the IK Adjust** ---
Weapon IK Adjust List: None (V Weapon IK Adjust List)

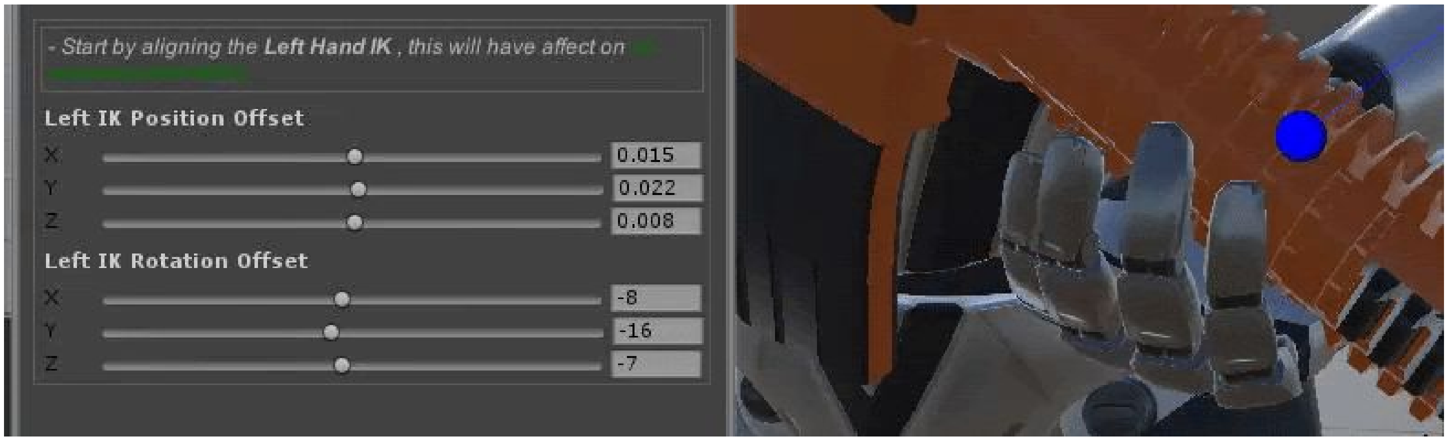
A new IK Adjustment List will be created and automatically assign for you, a new button called “**Edit IK Adjust List**” will appear, click on it and a new window will open, you can dock this window anywhere you like.

Hit **Play** and select the **ShooterController** to see this menu:


vShooterController_VBOT2.0_Custom@IKA
STATE : STANDING / RIFLE CATEGORY
Lock Aim
Lock Crouch
Lock Camera
CREATE NEW IK ADJUST
CHOOSE IK ADJUST

This weapon doesn't have a IKAdjust for the 'Rifle' category, click in the button below to create one.

Hit “**Create New IK Adjust**” and before we started playing around with the many cool IK options, we first need to **align our LeftHand IK**, you can do this by sliding the **XYZ Left IK Rotation and Position Offsets**.



- Start by aligning the **Left Hand IK** , this will have affect on

Left IK Position Offset

X 0.015
Y 0.022
Z 0.008

Left IK Rotation Offset

X -8
Y -16
Z -7

As you can see in the infobox this offsets **will affect all weapons** just like the Handler position/rotation you previously set up. Now that we have a base alignment we can proceed to explore the **features of IK Adjust**.

Once aligned, you don’t need to change the values of the LeftIK again since you now be able to create adjustments for each weapon and state.

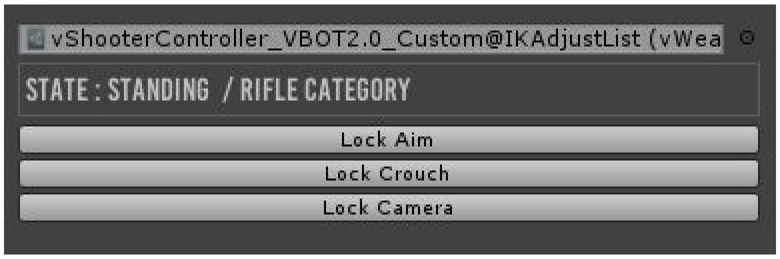
IK ADJUST FOR FINE TUNING OR CREATE NEW POSES

Now that we properly align the **defaultHandler** and the **LeftHandIK** for **all weapon prefabs**, we can start modifying the IK to create more **polished** or **even entirely new poses**, **for each state and each weapon category.**

The "State" will inform what state you are, there are basically 4 states that you can create unique IK modifications, those are:

- Standing
- Standing Crouch
- Aiming
- Aiming Crouch

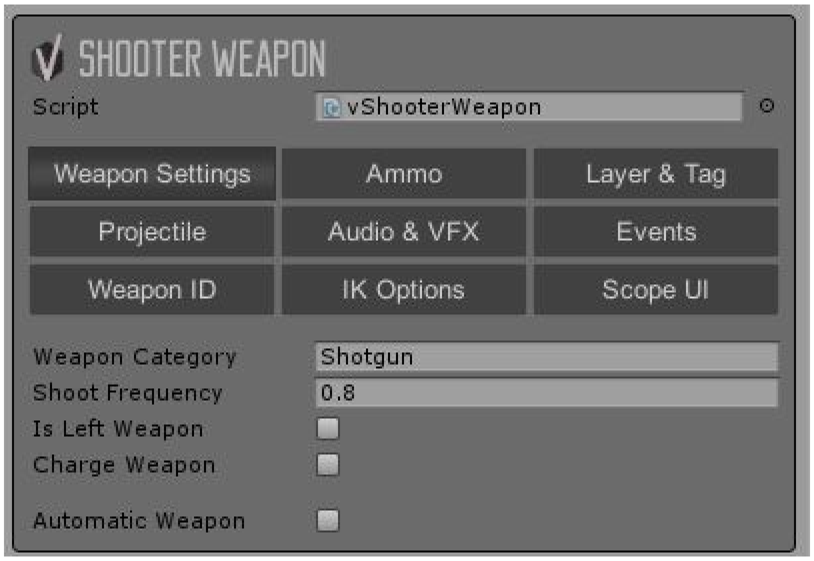
You can switch the state by pressing the buttons to LockAim, LockCrouch and LockCamera.
Start by Locking the Camera so that the Headtrack doesn't influence your pose.


vShooterController_VBOT2.0_Custom@IKAdjustList (vWea

STATE : STANDING / RIFLE CATEGORY

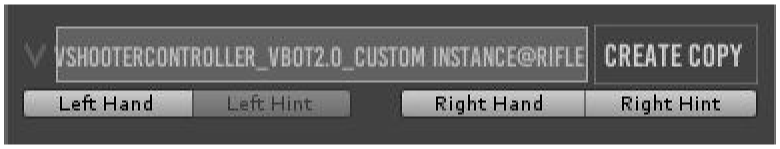
You can create different IKAdjusts for each weapon or use the same adjust if the weapon share the same **Category**, for example, if you have a large number of Pistols in your game that use the same pose, you don't need to create an IKAdjust for each weapon, just set the same Category for all the ShooterWeapon prefabs.

You can type any **Category** directly in the **ShooterWeapon** prefab:

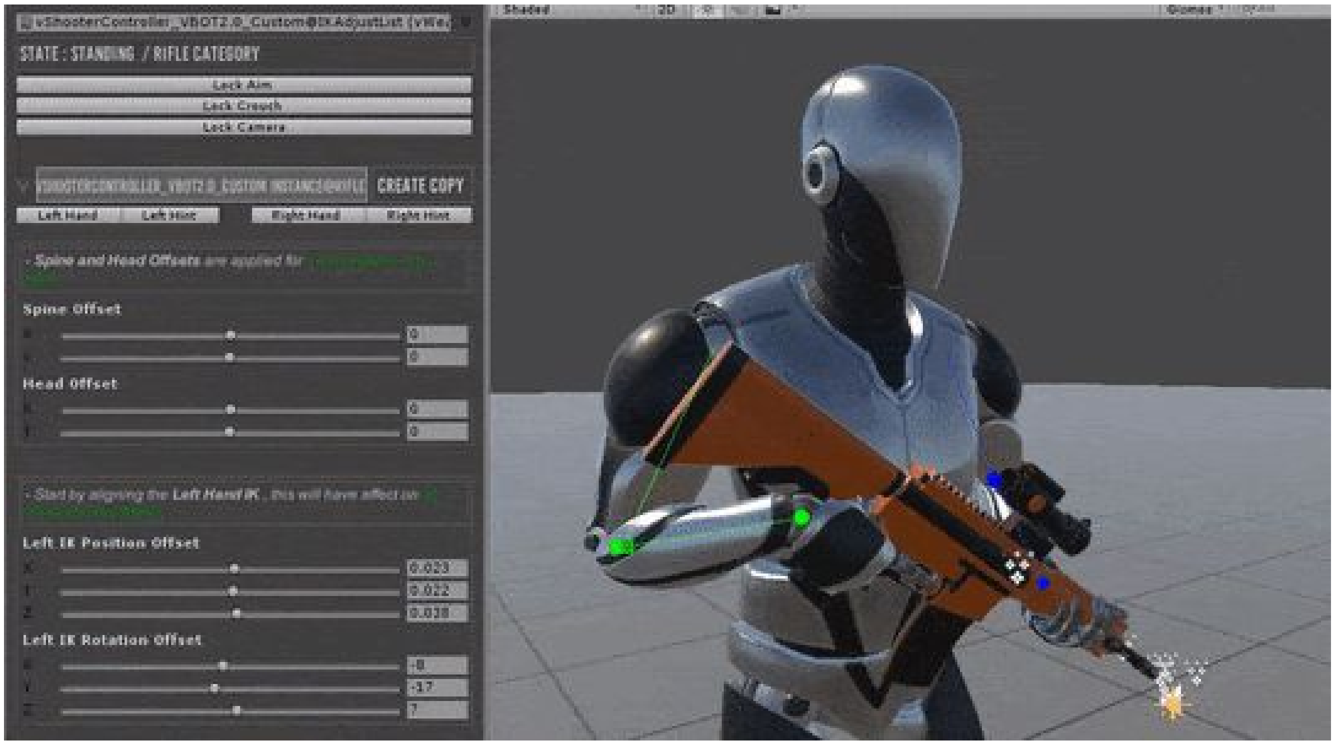


Script: vShooterWeapon
Weapon Category: Shotgun
Shoot Frequency: 0.8
Is Left Weapon: ☐
Charge Weapon: ☐
Automatic Weapon: ☐

You can create a **Copy** from another **IK Adjust**, for example if you have **2 rifles slightly different from each other**, just make a copy and do the adjustment for the other rifle.



You can also select the gizmos handlers directly from the menu or click on the little sphere/square of each arm. Or reset the position/rotation by clicking in their respect buttons, the “Reset” button will reset both.


vShooterController_VBOT2.0_Custom@IKAdjustList (vWe
STATE : STANDING / RIFLE CATEGORY
Lock Aim
Lock Crouch
Lock Camera
VSHOOTERCONTROLLER_VBOT2.0_CUSTOM INSTANCE@RIFLE
CREATE COPY
Left Hand
Left Hint
Right Hand
Right Hint
Shaded
2D
Gizmos

- Spine and Head Offsets are applied for [illegible]

Spine Offset
X 0
Y 0

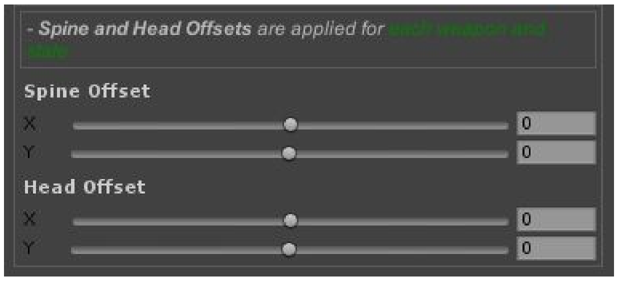
Head Offset
X 0
Y 0

- Start by aligning the Left Hand IK , this will have affect on [illegible]

Left IK Position Offset
X 0.023
Y 0.022
Z 0.028

Left IK Rotation Offset
X -8
Y -17
Z 7

You can also create **Offsets** for the **Spine** and **Head** to help you set up the position.



- Spine and Head Offsets are applied for [illegible]

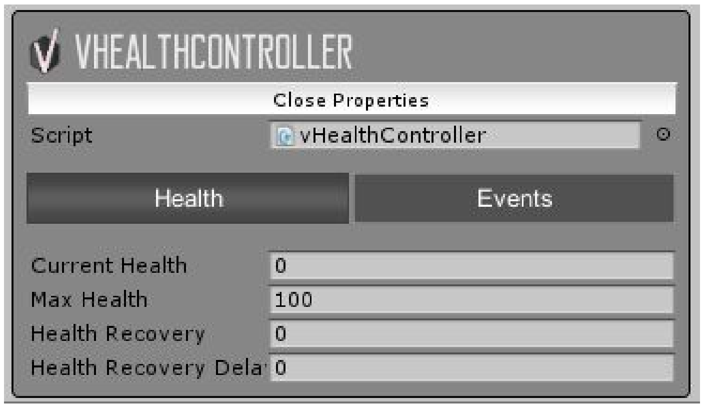
Spine Offset
X 0
Y 0

Head Offset
X 0
Y 0

- ***This feature was designed to help you better align the weapon into your Character Pose, but if you want the best result as possible the ideal is to replace the animations to animations created using your Character Rig. We recommend the use of uMotion to quickly and easily adjust the animations to your character’s rig.***

HOW TO APPLY DAMAGE TO A TARGET

The target must have a vHealthController and a Capsule Collider so that our Damage System can identify it as a living target and actually apply damage to it.

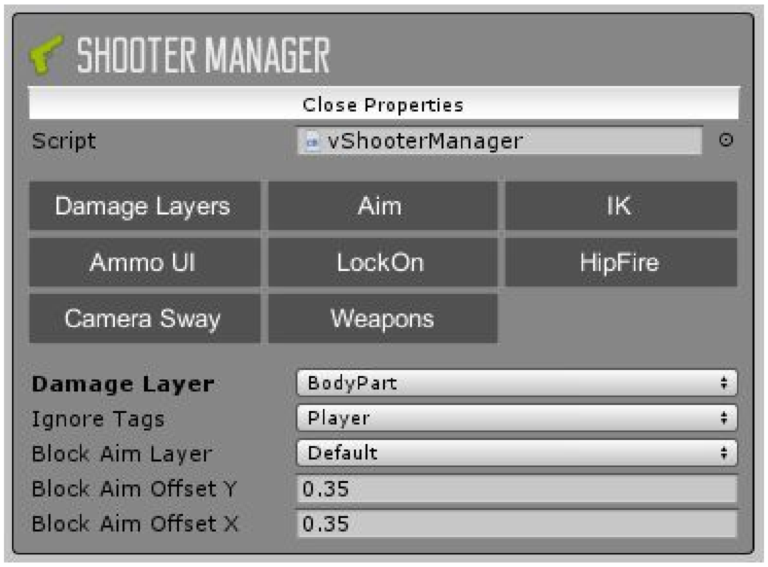


Script	vHealthController
Current Health	0
Max Health	100
Health Recovery	0
Health Recovery Dela	0

Shooter: You need to set the DamageLayer of your target in the ShooterManager.

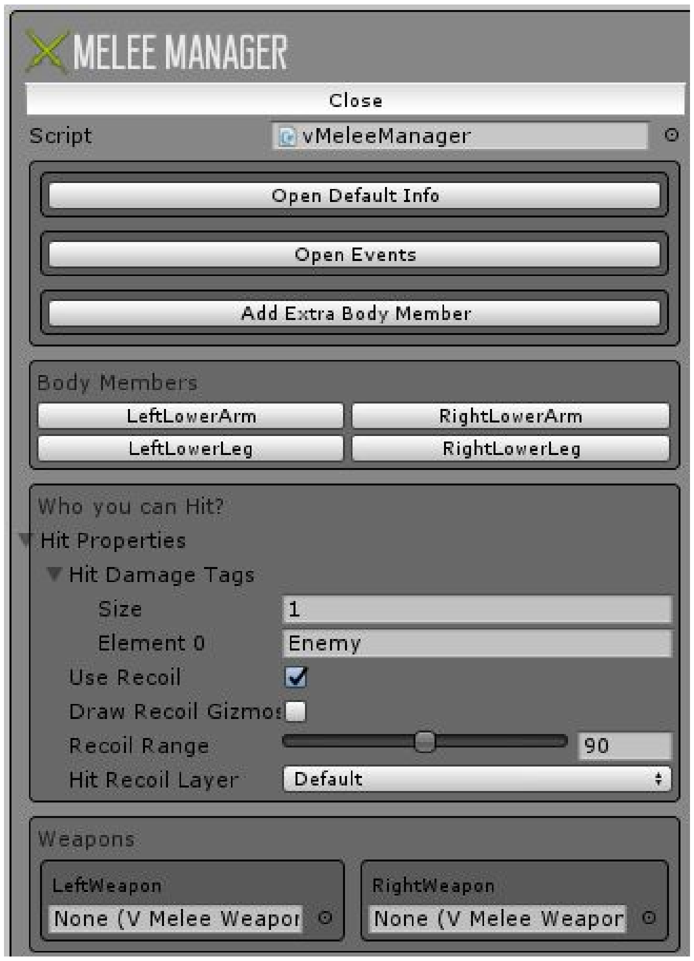
If you want to apply damage to individual body parts you can create a Ragdoll and set the collider layers to BodyPart, each ragdoll collider comes with a DamageReceiver, you can even set a Damage Multiplier if you want to apply extra damage in the Head for example.

OR if you're creating a simple top down game for example, you can save performance leaving the enemies without a ragdoll and applying damage directly to the main capsule collider, in this case you can set the Damage Layer to Enemy.



Script	vShooterManager
Damage Layer	BodyPart
Ignore Tags	Player
Block Aim Layer	Default
Block Aim Offset Y	0.35
Block Aim Offset X	0.35

MeleeCombat: You need to set the Tag of your Target in the MeleeManager / HitDamageTags field, you can assign several tags to hit different targets.


MELEE MANAGER

Close

Script: vMeleeManager

Open Default Info

Open Events

Add Extra Body Member

Body Members

LeftLowerArm | RightLowerArm

LeftLowerLeg | RightLowerLeg

Who you can Hit?

Hit Properties

Hit Damage Tags

Size	1
Element 0	Enemy
Use Recoil	✓
Draw Recoil Gizmos	
Recoil Range	90
Hit Recoil Layer	Default

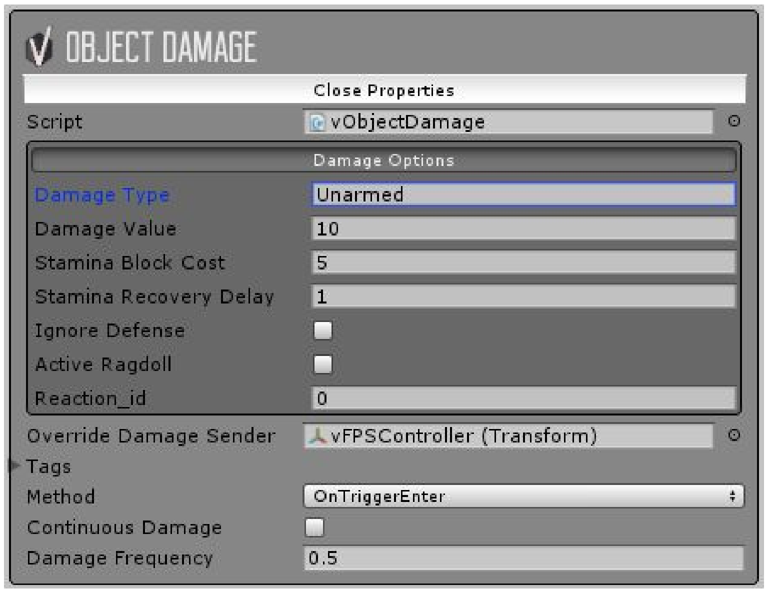
Weapons

LeftWeapon: None (V Melee Weapon

RightWeapon: None (V Melee Weapon

HOW TO APPLY DAMAGE TO THE PLAYER

We have a few examples on how to apply damage to the Controller, basically you need the **vObjectDamage** script which is attached to the Pendulum and Spikes.

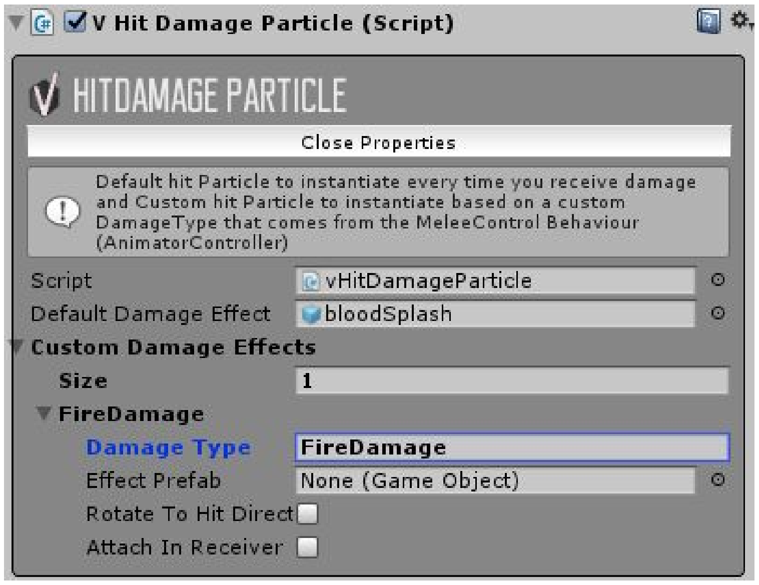

OBJECT DAMAGE

Close Properties

Script	vObjectDamage
Damage Options	
Damage Type	Unarmed
Damage Value	10
Stamina Block Cost	5
Stamina Recovery Delay	1
Ignore Defense	☐
Active Ragdoll	☐
Reaction_id	0
Override Damage Sender	vFPSController (Transform)
Tags	
Method	OnTriggerEnter
Continuous Damage	☐
Damage Frequency	0.5

- **DamageType**: Used together with the HitParticleDamage component, you can trigger different particles for different type of damage.

For example if you have an area with fire, you can add a vObjectDamage there and add a DamageType of "FireDamage", then add a Custom Damage Effect with the same DamageType to your **HitDamageParticle** on your Character and add the particle effect to burn your character.


V Hit Damage Particle (Script)

HITDAMAGE PARTICLE

Close Properties

Default hit Particle to instantiate every time you receive damage and Custom hit Particle to instantiate based on a custom DamageType that comes from the MeleeControl Behaviour (AnimatorController)

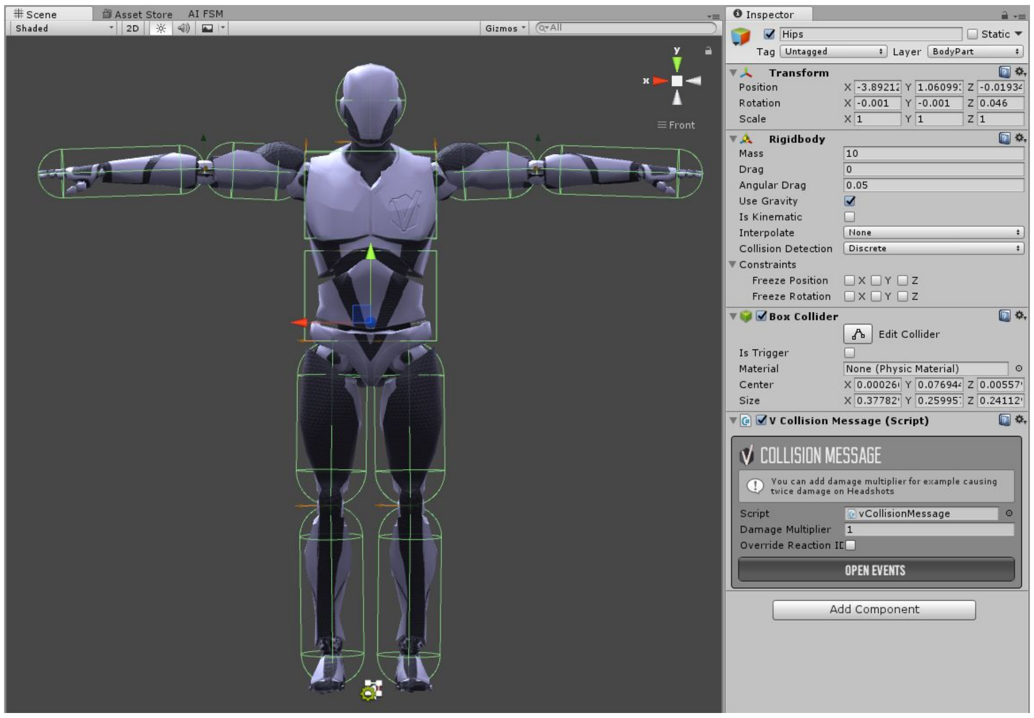
Script	vHitDamageParticle
Default Damage Effect	bloodSplash
Custom Damage Effects	
Size	1
FireDamage	
Damage Type	**FireDamage**
Effect Prefab	None (Game Object)
Rotate To Hit Direct	☐
Attach In Receiver	☐

*This component is attached to the Controller or any object that contains the HealthController

- **DamageValue**: How much damage it will be applied to the vHealthController of the target.
- **Stamina Block Cost**: You can ignore that option, it's only for the ThirdPersonController.
- **Stamina Recovery Delay**: You can ignore that option, it's only for the ThirdPersonController.
- **Ignore Defense:** If you're using a MeleeCombat Controller, it will ignore the defense and apply damage anyways.
- **Active Ragdoll**: It will activate the ragdoll on your character, if it has one.
- **Reaction ID**: You can trigger specific hit reaction animation, you can use -1 if you don't want to trigger any animation.
- **Override Damage Sender**: Assign the root object otherwise the AI will target the object that has this component instead. For Example: If you apply the vObjectDamage to be a Hitbox of a LeftHand of a character, the vHealthController or AI will have the LeftHand as the target instead of the GameObject parent.
- **Tags**: What tags you will apply damage to
- **Method**: OnTriggerEnter or OnCollisionEnter
- **Continuous Damage**: Useful for fire damage for example
- **Damage Frequency**: Frequency to apply the damage, if Continuous Damage is enabled.

Using the Ragdoll Colliders as BodyParts to inflict precise damage.

SHOOTER > If you want to cause damage for each body member using the ragdoll colliders, UNCHECK the “Disable Colliders” and you can add damage multiplier on each member.



Hips
Static
Tag Untagged Layer BodyPart

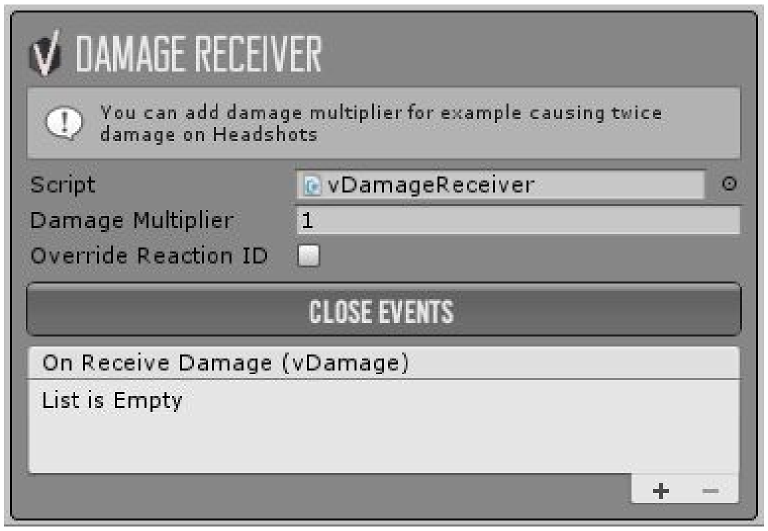
Transform
Position X -3.89212 Y 1.06099 Z -0.01934
Rotation X -0.001 Y -0.001 Z 0.046
Scale X 1 Y 1 Z 1

Rigidbody
Mass 10
Drag 0
Angular Drag 0.05
Use Gravity ✓
Is Kinematic ☐
Interpolate None
Collision Detection Discrete
Constraints
Freeze Position ☐ X ☐ Y ☐ Z
Freeze Rotation ☐ X ☐ Y ☐ Z

Box Collider
Is Trigger ☐
Material None (Physic Material)
Center X 0.00026 Y 0.07694 Z 0.00557
Size X 0.37782 Y 0.25995 Z 0.24112

V Collision Message (Script)
You can add damage multiplier for example causing twice damage on Headshots
Script vCollisionMessage
Damage Multiplier 1
Override Reaction ID ☐

* You must use a different Layer in the Ragdoll Colliders like “BodyPart” and another for the main capsule collider like "Enemy", this way the Detection will detect the Enemy object as a target, but the ShooterManager will actually apply damage to the "BodyPart".



You can add damage multiplier for example causing twice damage on Headshots

Script: vDamageReceiver
Damage Multiplier: 1
Override Reaction ID: ☐

On Receive Damage (vDamage)
List is Empty

Body Parts use the Damage Receiver to receive damage.

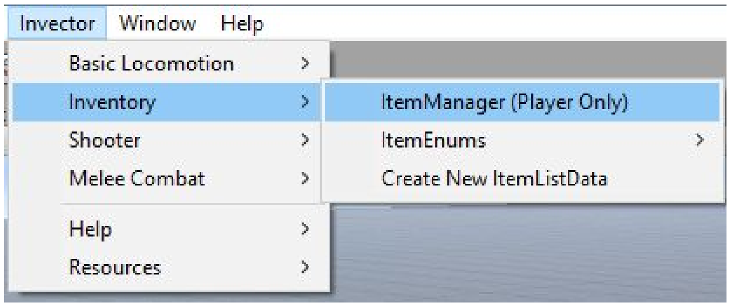
A DamageReceiver is attached to each ragdoll collider, this will allow to Player or AI to apply damage to each bodypart instead of the CapsuleCollider.

- **Damage Multiplier**: multiply the damage value
- **Override a Reaction ID**: Check this option to override the hit reaction animation to trigger a specific animation.

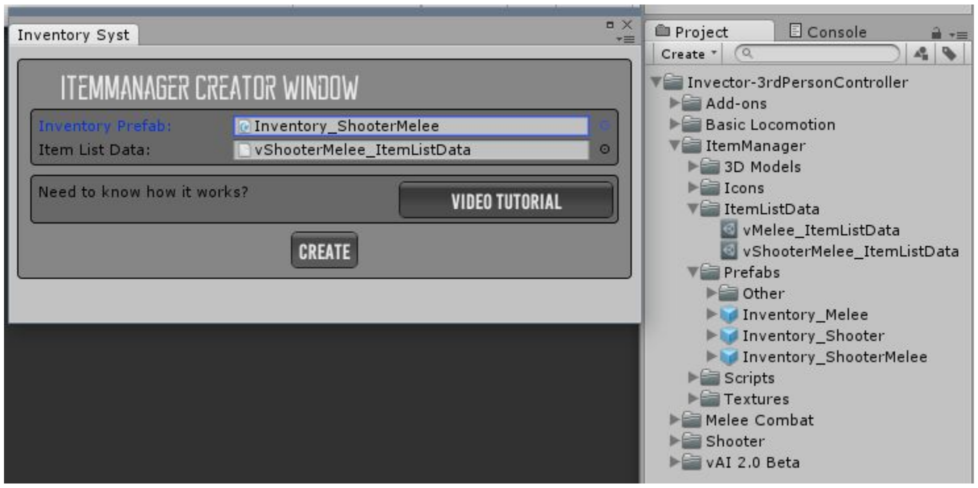
For example, if you want to cause 2x damage and trigger a specific reaction animation when shooting in the Head, simply change the values in this component.

ITEM MANAGER

- Add the ItemManager into your Player from the menu **Invector > Inventory > ItemManager**

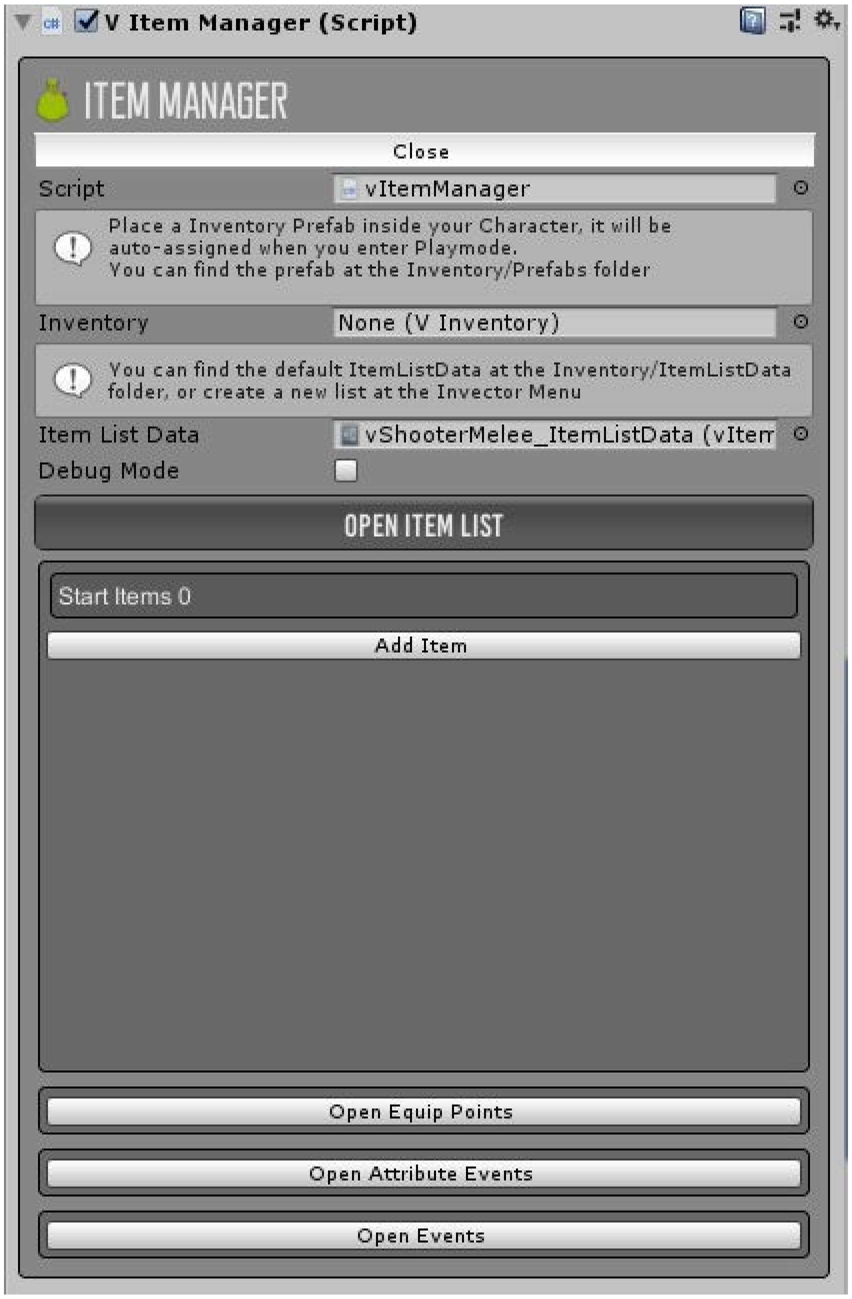


- Select the Inventory Prefab from the **Project > ItemManager > Prefabs then drag and drop inside your Third Person Controller, when you hit play it will be automatically assigned to the ItemManager.**
- and a **ItemListData** > **vShooterMelee_ItemListData**



Inventory Prefab: Inventory_ShooterMelee
Item List Data: vShooterMelee_ItemListData

- You can also add the ItemManager manually using the Add Component button via inspector, but don't forget to assign an ItemListData.


V Item Manager (Script)

ITEM MANAGER

Close

Script: vItemManager

Place a Inventory Prefab inside your Character, it will be auto-assigned when you enter Playmode.
You can find the prefab at the Inventory/Prefabs folder

Inventory: None (V Inventory)

You can find the default ItemListData at the Inventory/ItemListData folder, or create a new list at the Invector Menu

Item List Data: vShooterMelee_ItemListData (vItem

Debug Mode: ☐

OPEN ITEM LIST

Start Items 0

Add Item

Open Equip Points

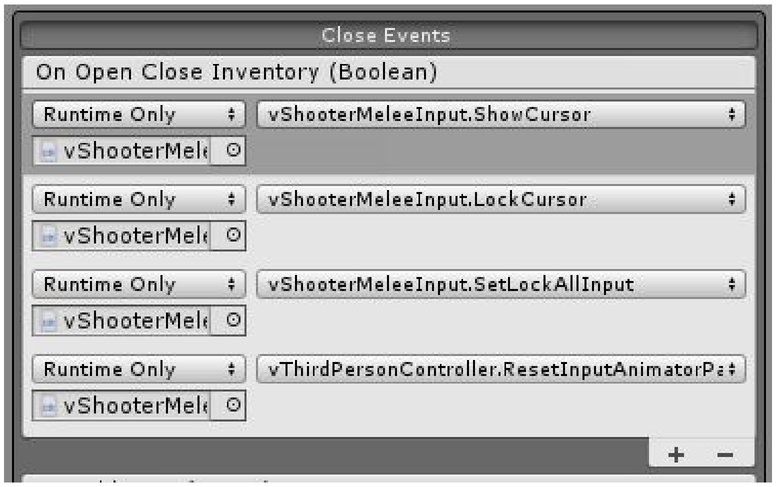
Open Attribute Events

Open Events

In the tab **Events** you can call methods like lock the input of the character while the Inventory is Open, just assign the Character (from the scene hierarchy) and call the method LockInput from the vMeleeCombatInput or vShooterMeleeInput.

Here are the most used events you can use to:

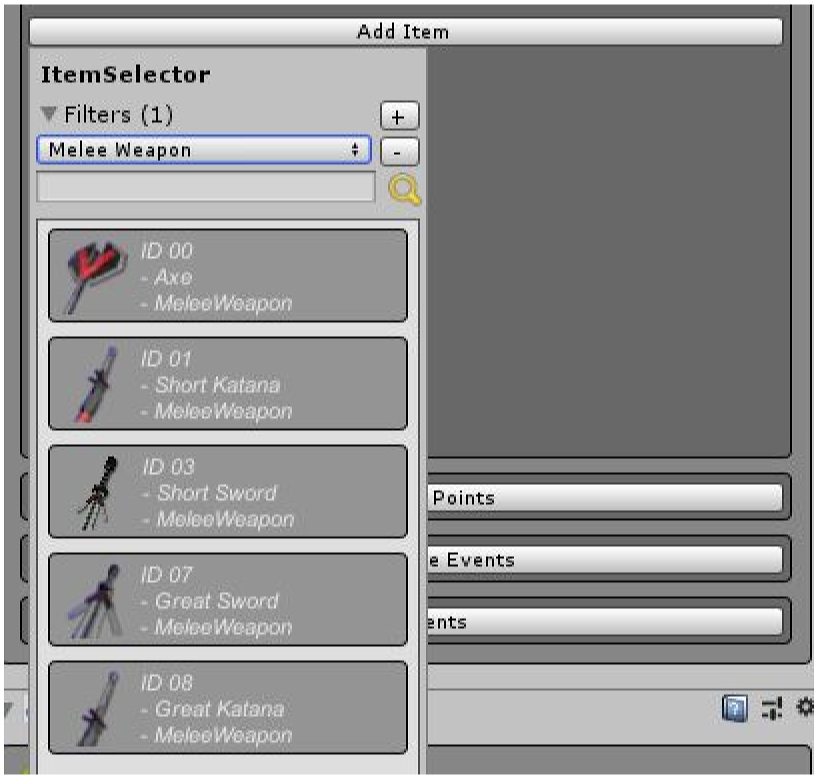
- Hide/Show the mouse cursor
- Lock/Unlock the cursor at the screen center
- SetLockAllInput to lock all the input from the controller (basic, melee and shooter)
- ResetInputAnimatorParameters to reset back to zero the animator parameters, usefull when using the CharacterCameraPreview inside the Inventory, this way if you're walking and open the inventory, there character will be at Idle in the camera preview.



On Open Close Inventory (Boolean)

Runtime Only	vShooterMeleeInput.ShowCursor
vShooterMel	
Runtime Only	vShooterMeleeInput.LockCursor
vShooterMel	
Runtime Only	vShooterMeleeInput.SetLockAllInput
vShooterMel	
Runtime Only	vThirdPersonController.ResetInputAnimatorPa
vShooterMel	

Click at the Add Item button to open the Filter, so you can search or filter exactly what ItemType you need: You can add multiple ItemTypes to filter multiple items.



ID 00
- Axe
- MeleeWeapon

ID 01
- Short Katana
- MeleeWeapon

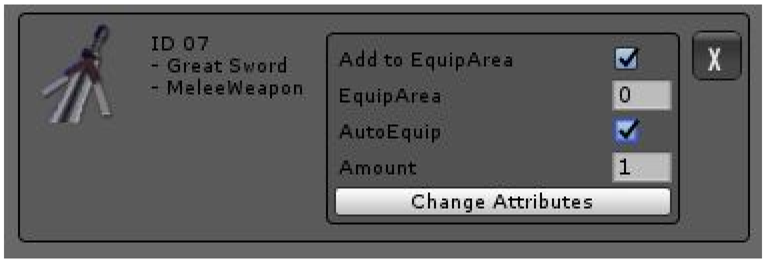
ID 03
- Short Sword
- MeleeWeapon

ID 07
- Great Sword
- MeleeWeapon

ID 08
- Great Katana
- MeleeWeapon

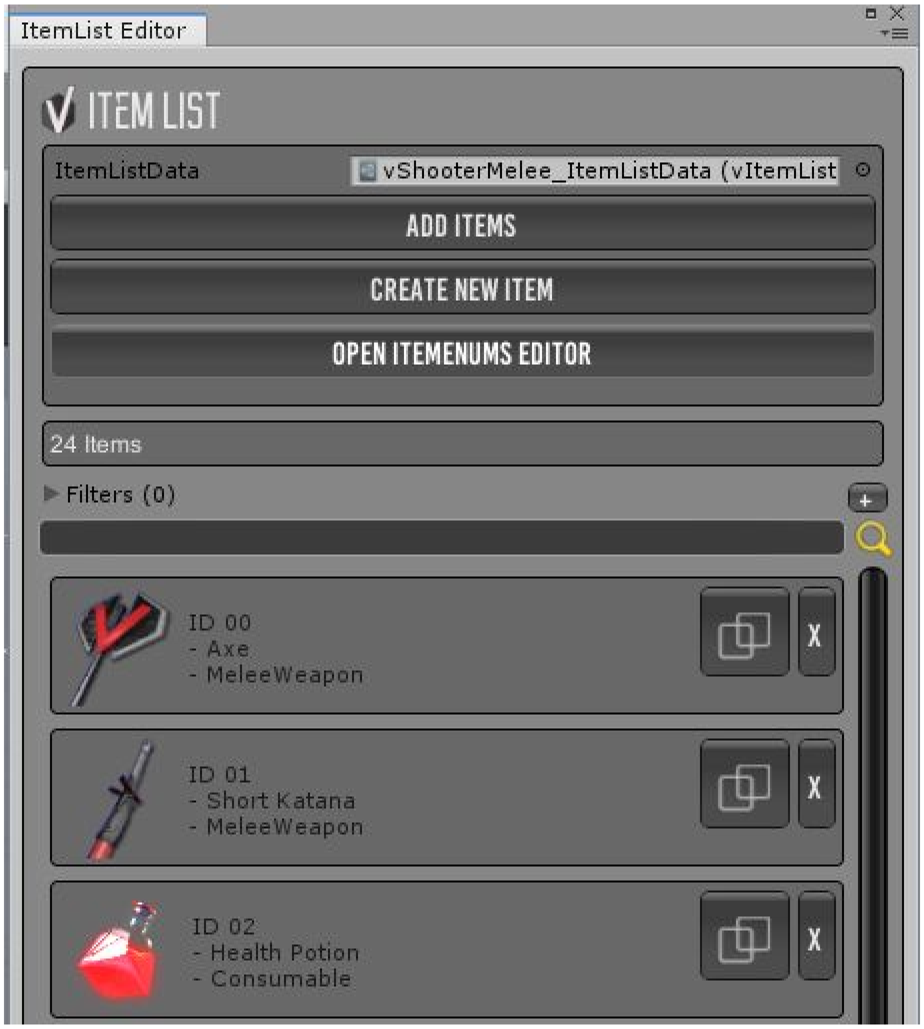
Once you add a Item in the list, you can use the option **Add To EquipArea** to automatically assign this item to a specific EquipArea (it will only be assigned if your itemSlots are currently empty, if one is already assigned, it will be equipped in the next one in the list)

The **AutoEquip** will auto equip this item to the EquipArea and also change the EquipmentDisplayWindow to where the Item is equipped.


ID 07
- Great Sword
- MeleeWeapon

Add to EquipArea ✓
EquipArea 0
AutoEquip ✓
Amount 1

Click in the **Open Item List** button, to manage, search or create new items


ItemList Editor

ITEM LIST

ItemListData vShooterMelee_ItemListData (vItemList

24 Items

Filters (0)

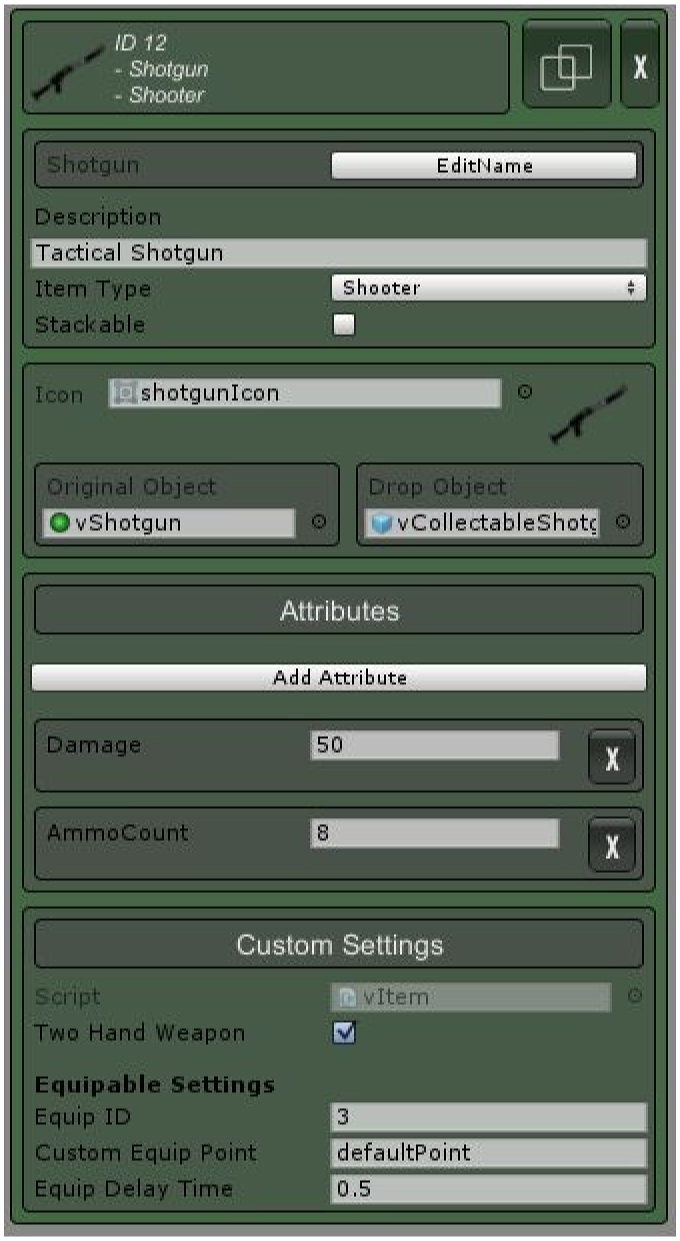
ID 00
- Axe
- MeleeWeapon

ID 01
- Short Katana
- MeleeWeapon

ID 02
- Health Potion
- Consumable

You can create new items or duplicate a current one, keep in mind that each item has a unique ID.

When creating a Weapon Item, you need to assign the **Original Object** (that instantiate into the Player with a vMeleeWeapon or vShooterWeapon) and a **DropObject** which we have a prefab called "**CollectableEquipment**" that you can use and it will automatically drop the item you assign or create a unique collectable with a mesh that matches your item.



ID 12
- Shotgun
- Shooter

Shotgun EditName

Description
Tactical Shotgun

Item Type	Shooter
Stackable	☐
Icon	shotgunIcon
Original Object	vShotgun
Drop Object	vCollectableShotg

Attributes

Add Attribute

Damage	50
AmmoCount	8

Custom Settings

Script	vItem
Two Hand Weapon	✓

Equipable Settings

Equip ID	3
Custom Equip Point	defaultPoint
Equip Delay Time	0.5

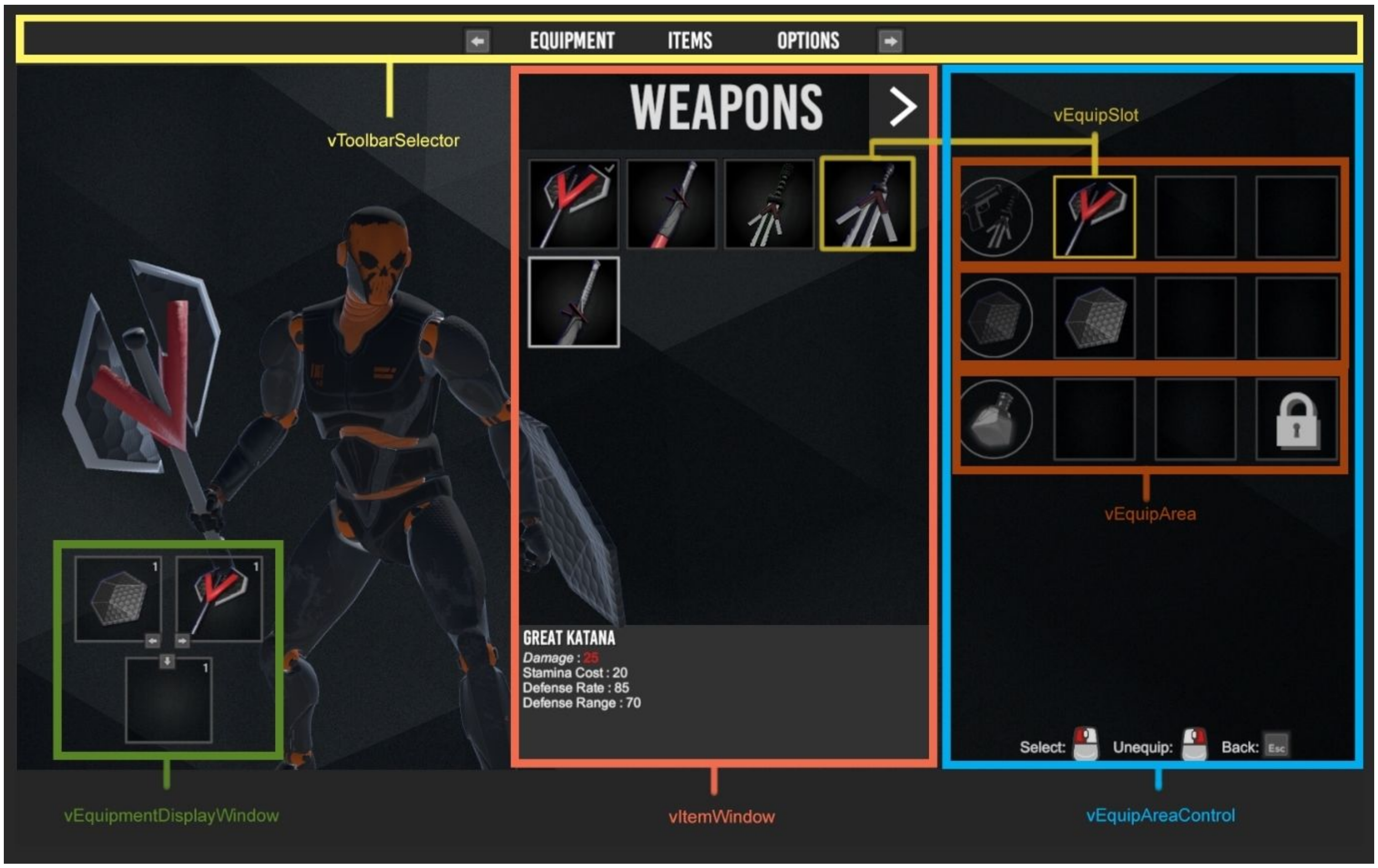
Don't forget to add the attribute Damage & AmmoCount of your weapon, this will allow you to drop and collect your weapon with the same amount of weapon, making it into a unique weapon.

This Inventory Example goes further and further into options to customize, like consumable items, if is stackable or not, and much more that is better explained on video tutorials that you can watch on our Youtube Channel.

INVENTORY

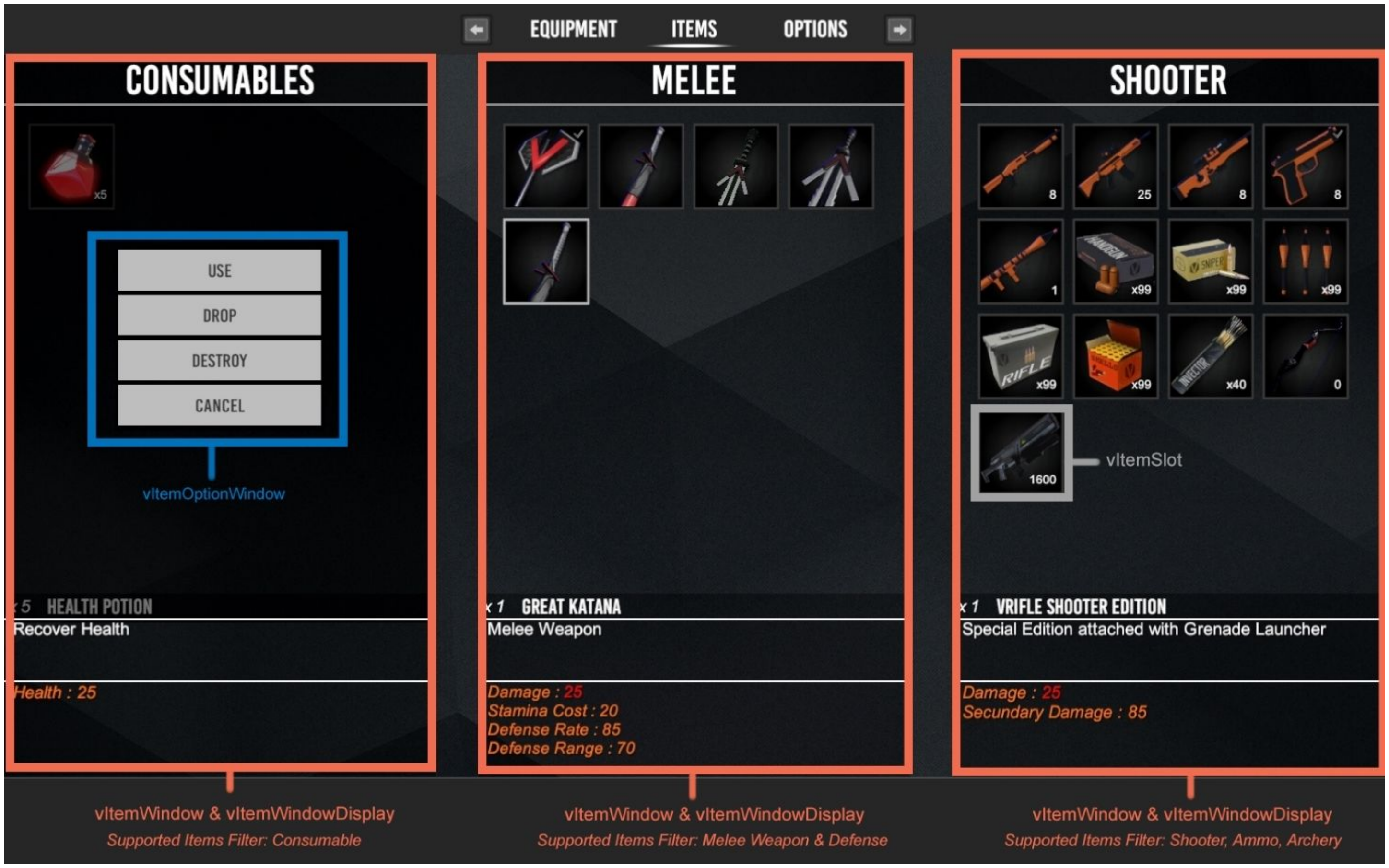
We have basically 3 Inventory Examples in the project, you can find them in the folder ItemManager/Prefabs. Just drag and drop inside your character to test it, make sure to also use the corresponding ItemListData at your ItemManager according to what Inventory you're using.

For ex: ShooterOnly requires the vShooterMelee_ItemListData, MeleeOnly requires the vMelee_ItemListData



- **ToolbarSelector**: It's basically a menu, you can set the input to switch tabs at your Inventory
- **Equipment Display Window**: It shows the current slot of a EquipArea, you can have multiple slots and multiple EquipAreas at the same time and rotate the slots via input
- **EquipArea**: A row of EquipSlots
- **Equip Area Control**: Responsible to inform the EquipmentDisplay
- **EquipSlots**: Display a specific Item, you can use the ItemType to filter what item can be displayed on this slot
- **ItemWindow**: A window that display Items, you can filter what ItemType will be displayed using the filter 'Supported Items'

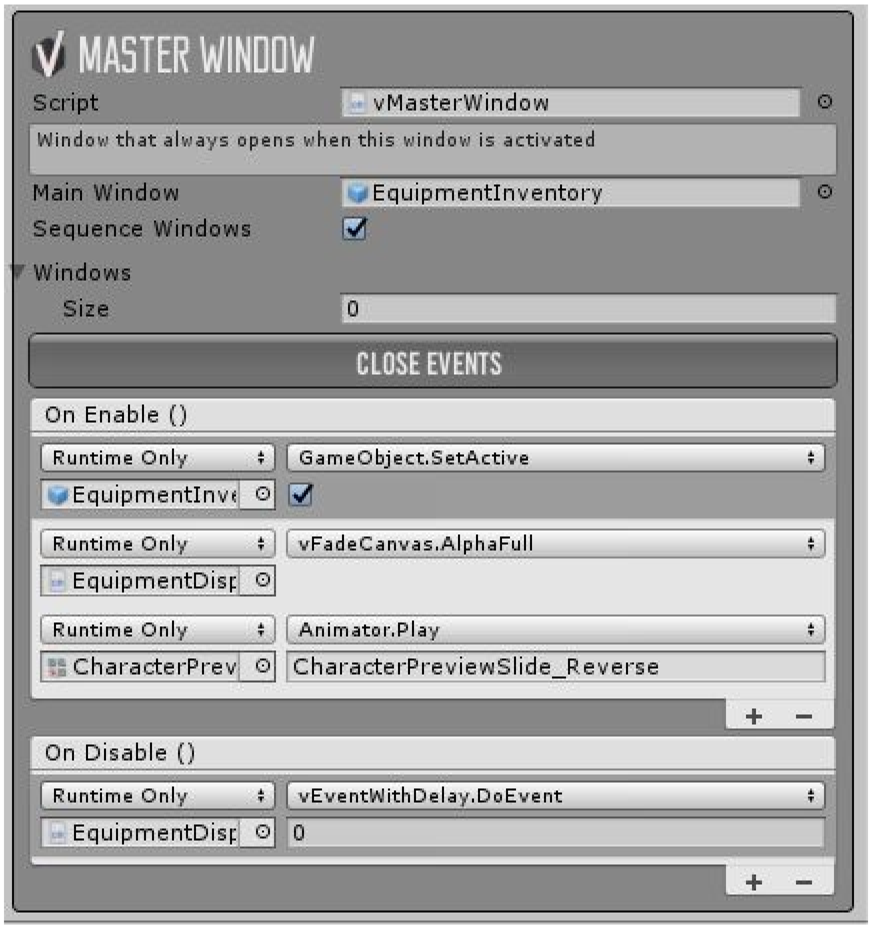
This is an example of a window that displays several different Items based on their Types.


EQUIPMENT
ITEMS
OPTIONS
CONSUMABLES
MELEE
SHOOTER
USE
DROP
DESTROY
CANCEL
vItemOptionWindow
vItemSlot
x 5 HEALTH POTION
Recover Health
Health : 25
x 1 GREAT KATANA
Melee Weapon
Damage : 25
Stamina Cost : 20
Defense Rate : 85
Defense Range : 70
x 1 VRIFLE SHOOTER EDITION
Special Edition attached with Grenade Launcher
Damage : 25
Secundary Damage : 85
vItemWindow & vItemWindowDisplay
Supported Items Filter: Consumable
vItemWindow & vItemWindowDisplay
Supported Items Filter: Melee Weapon & Defense
vItemWindow & vItemWindowDisplay
Supported Items Filter: Shooter, Ammo, Archery

- ItemWindowDisplay: Manage what happens with the ItemWindow when using, drop, destroy and cancel
- ItemOptionWindow: Buttons that buttons UI to use, equip, drop, destroy and cancel

You can see a **MasterWindow** component on each main window of the Inventory, you can use the MasterWindow to manage your windows for example if you open a window and open another, you can press the back button to close the current window and get back to the previous, it's basically a window manager.

You can also use the Events OnEnable/OnDisable to for example, turn on / off specific objects, call animations, sounds, etc...



Script: vMasterWindow

Window that always opens when this window is activated

Main Window: EquipmentInventory

Sequence Windows: ✓

Windows

Size: 0

On Enable ()

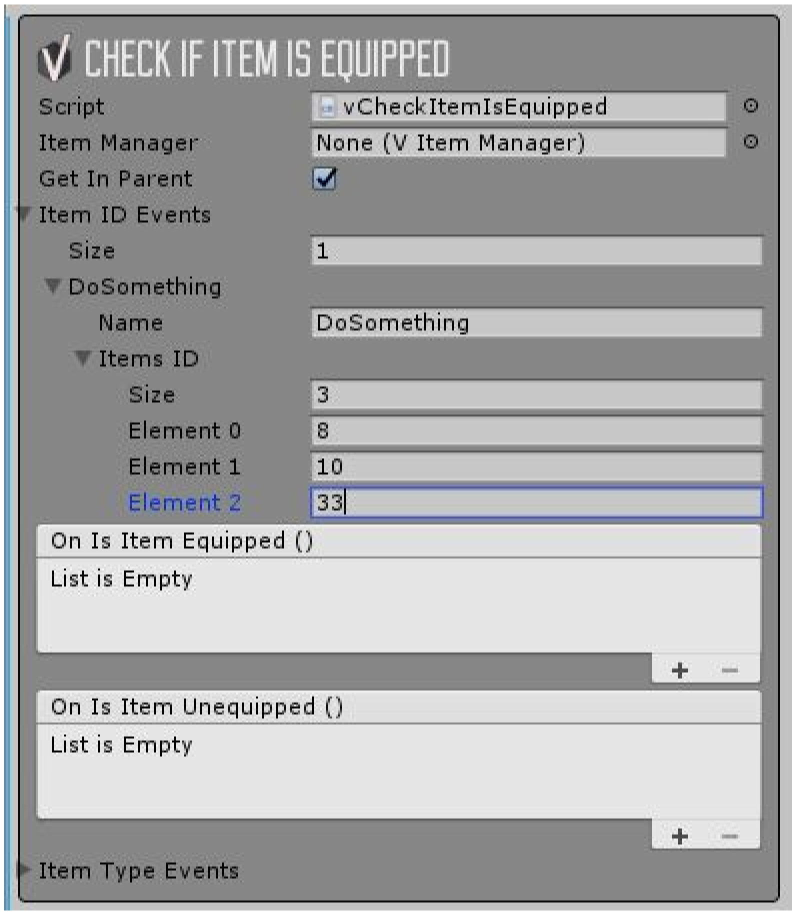
Runtime Only	GameObject.SetActive	
EquipmentInve	✓	
Runtime Only	vFadeCanvas.AlphaFull	
EquipmentDisp		
Runtime Only	Animator.Play	
CharacterPrev	CharacterPreviewSlide_Reverse	

On Disable ()

Runtime Only	vEventWithDelay.DoEvent
EquipmentDisp	0

CHECK IF ITEM IS EQUIPPED

You can use the component *vCheckItemIsEquipped* to check if a specific Item or ItemType is currently equipped.



CHECK IF ITEM IS EQUIPPED

Script	vCheckItemIsEquipped
Item Manager	None (V Item Manager)
Get In Parent	✓
Item ID Events	
Size	1
DoSomething	
Name	DoSomething
Items ID	
Size	3
Element 0	8
Element 1	10
Element 2	33

On Is Item Equipped ()
List is Empty

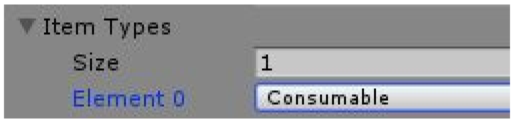
On Is Item Unequipped ()
List is Empty

Item Type Events

You can attach this component directly to the character or inside the character (check the option “GetInParent” if it’s inside the Player).

Name - just the name of your event, so it’s easier to identify what it does
ItemsID - You can add one or more items to trigger something specific, for example, when equipped with Potions, you can enable the UI Button to use those items, when unequipped simply disable the object.

You could either filter the ItemID’s of your Potions or set the ItemType Consumable in this example.



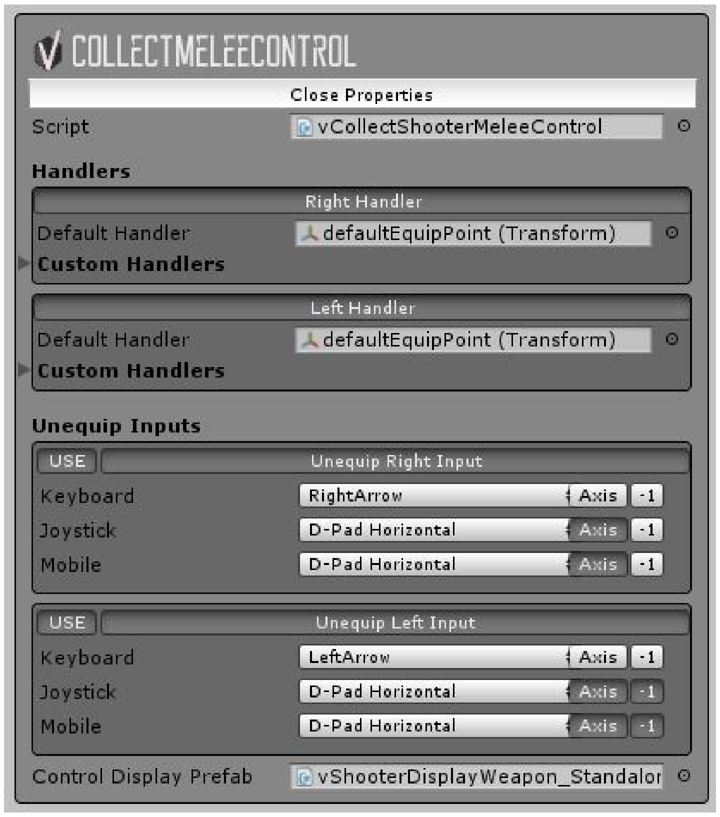
Item Types	
Size	1
Element 0	Consumable

COLLECTABLE STANDALONE (NO INVENTORY)

If you don’t want to use the ItemManager to manage your items, we have another solution for ‘on demand’ collectibles, notice that you can only equip 1 item, once you try to equip another the current item will drop.

Take a look into the Demo Scene called “vShooterMelee_NOInventory”, instead of adding the ItemManager component, now you will add the “vCollectShooterMeleeControl” component to automatically collect and equip weapons.

You need to create the defaultEquipPoint to equip weapons and assign inputs to drop them.



COLLECTMELEECONTROL

Close Properties

Script: vCollectShooterMeleeControl

Handlers

Right Handler
Default Handler: defaultEquipPoint (Transform)
Custom Handlers

Left Handler
Default Handler: defaultEquipPoint (Transform)
Custom Handlers

Unequip Inputs

USE — Unequip Right Input

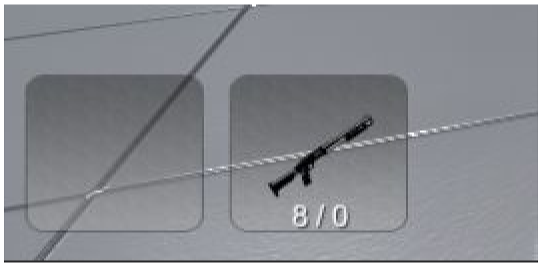
Keyboard	RightArrow	Axis	-1
Joystick	D-Pad Horizontal	Axis	-1
Mobile	D-Pad Horizontal	Axis	-1

USE — Unequip Left Input

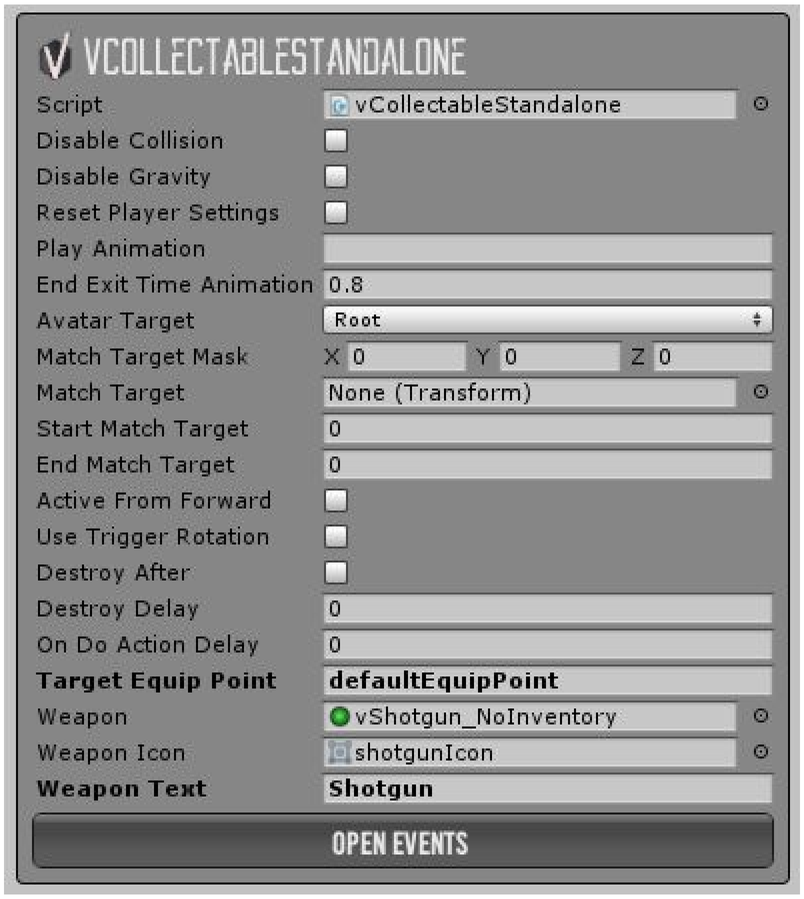
Keyboard	LeftArrow	Axis	-1
Joystick	D-Pad Horizontal	Axis	-1
Mobile	D-Pad Horizontal	Axis	-1

Control Display Prefab: vShooterDisplayWeapon_Standalor

We also have a pretty simple example of a Display HUD to show what weapons you’re equipped with, it’s call “vShooterDisplayWeapon”, search in the project folder and drag and drop the prefab into the scene.



For the ItemManager we need a prefab for the actual weapon that goes into the Player and another to be the Collectable, but in this case the CollectableStandalone is both. Take a look into one of the several example of collectables we have for both melee and shooter weapons.

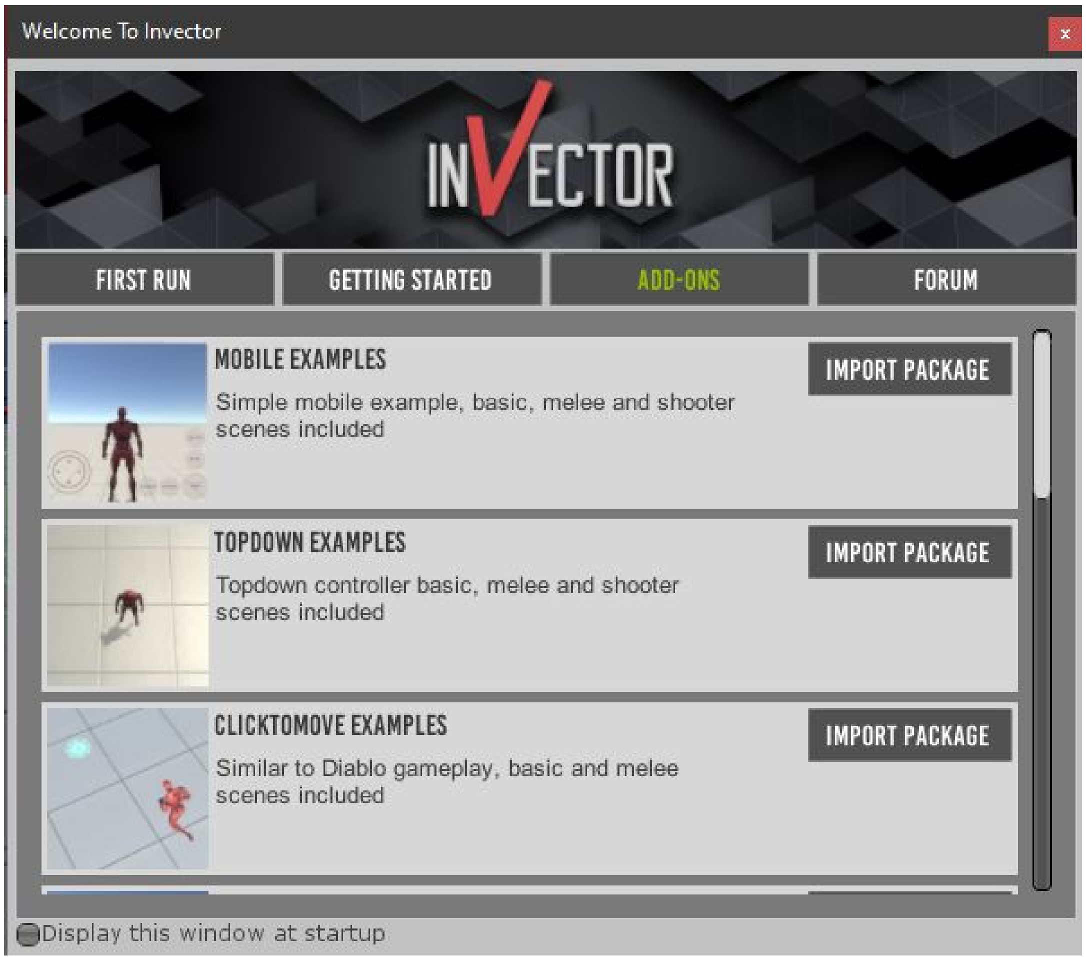

VCOLLECTABLESTANDALONE

Script	vCollectableStandalone
Disable Collision	☐
Disable Gravity	☐
Reset Player Settings	☐
Play Animation	
End Exit Time Animation	0.8
Avatar Target	Root
Match Target Mask	X 0 Y 0 Z 0
Match Target	None (Transform)
Start Match Target	0
End Match Target	0
Active From Forward	☐
Use Trigger Rotation	☐
Destroy After	☐
Destroy Delay	0
On Do Action Delay	0
Target Equip Point	**defaultEquipPoint**
Weapon	vShotgun_NoInventory
Weapon Icon	shotgunIcon
Weapon Text	**Shotgun**

It's important to assign the correct gameobjects into the Events, we turn off the collision and gravity of the weapons when equipped and turn on when you drop them.

TOPDOWN / 2.5D / POINT & CLICK CONTROLLER / MOBILE

We have different types of controllers that are included in the project as a bonus, you can import those packages by going to ***Invector > WelcomeWindow > Add-ons***



MOBILE EXAMPLES

Simple mobile example, basic, melee and shooter scenes included

TOPDOWN EXAMPLES

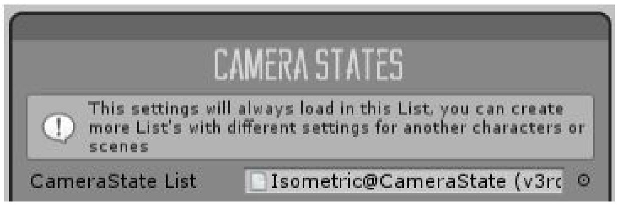
Topdown controller basic, melee and shooter scenes included

CLICKTOMOVE EXAMPLES

Similar to Diablo gameplay, basic and melee scenes included

Each scene contains examples on how to set up each gameplay type.

To turn your Third Person Controller into a TopDown or Isometric controller just go into your ThirdPersonCamera and change the CameraState to **TopDown@CameraState, Isometric@CameraState or 2.5@CameraState** depending on what controller you want.



This settings will always load in this List, you can create more List's with different settings for another characters or scenes

CameraState List Isometric@CameraState (v3rc

Go to the add-on folder you want and replace your current **vThirdPersonController** component for the **vTopDownController** or **2_5Dcontroller** in the Player Inspector.

To use the **Point&Click** you can still use the vThirdPersonController, but you will need to replace the Input to **vPointAndClickInput or vMeleePointClickInput** (if it's a melee character), for more information check the Invector_Point&Click_Melee.

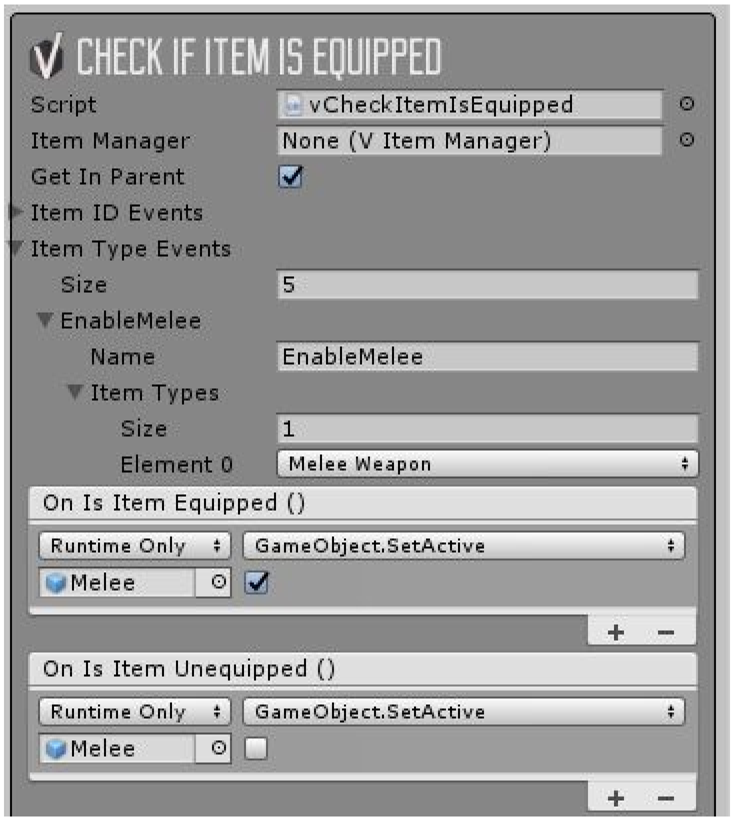
And for the **2.5DController** check the 2.5Demo scene, you will need a 2.5Path to navigate.

MOBILE CONTROLS

After importing the mobile package, you can create a regular ThirdPersonController using the Character Creator Window and after setting up the controller and aligning the handlers, search for the Mobile UI Controls prefabs:

MobileUI_ShooterMelee_Inventory - if you want to use Inventory
MobileUI_ShooterMelee_NoInventory - if you don't want to use Inventory, check the demo scenes to see how it works and choose one style for your game.

If you're using the Inventory you can use the component **vCheckItemIsEquipped** to turn on/off the mobile buttons:



Script	vCheckItemIsEquipped
Item Manager	None (V Item Manager)
Get In Parent	✓
Item ID Events	
Item Type Events	
Size	5
EnableMelee	
Name	EnableMelee
Item Types	
Size	1
Element 0	Melee Weapon

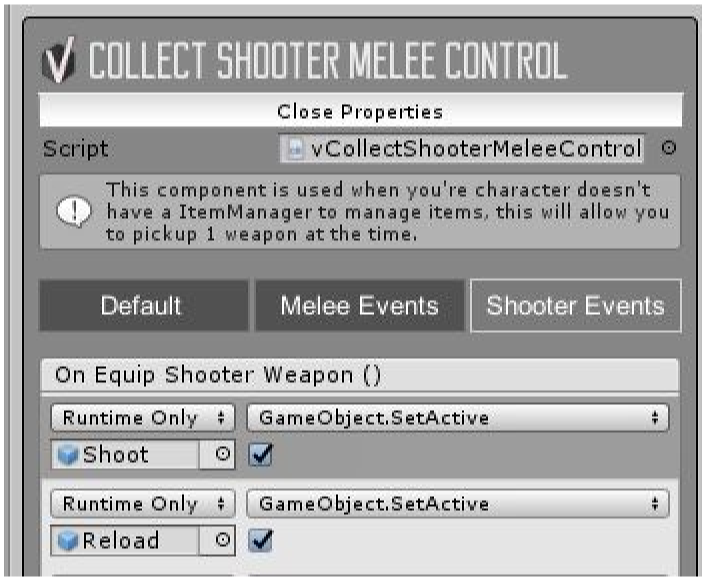
On Is Item Equipped ()

Runtime Only	GameObject.SetActive
Melee	✓

On Is Item Unequipped ()

Runtime Only	GameObject.SetActive
Melee	☐

If you’re not using the Inventory, you can use the **vCollectShooterMeleeControl** Events to manage the buttons, it identifies if you’re using a shooter weapon or a melee weapon only.



Script: vCollectShooterMeleeControl

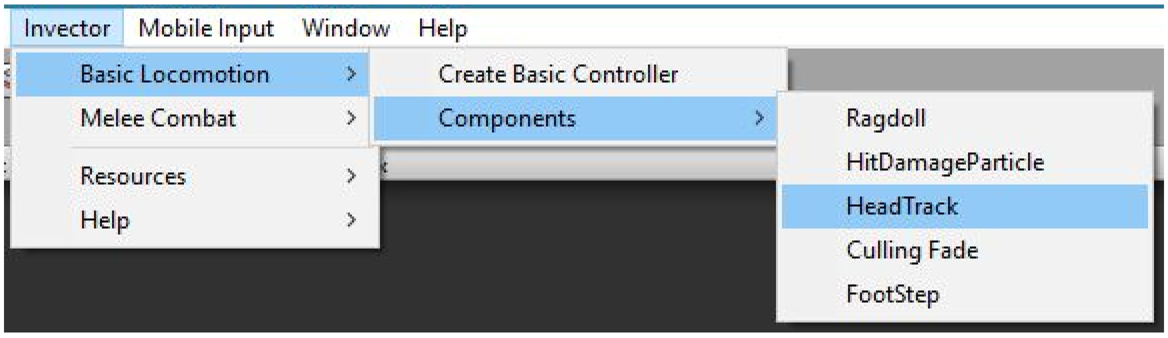
This component is used when you're character doesn't have a ItemManager to manage items, this will allow you to pickup 1 weapon at the time.

GameObject.SetActive ✓

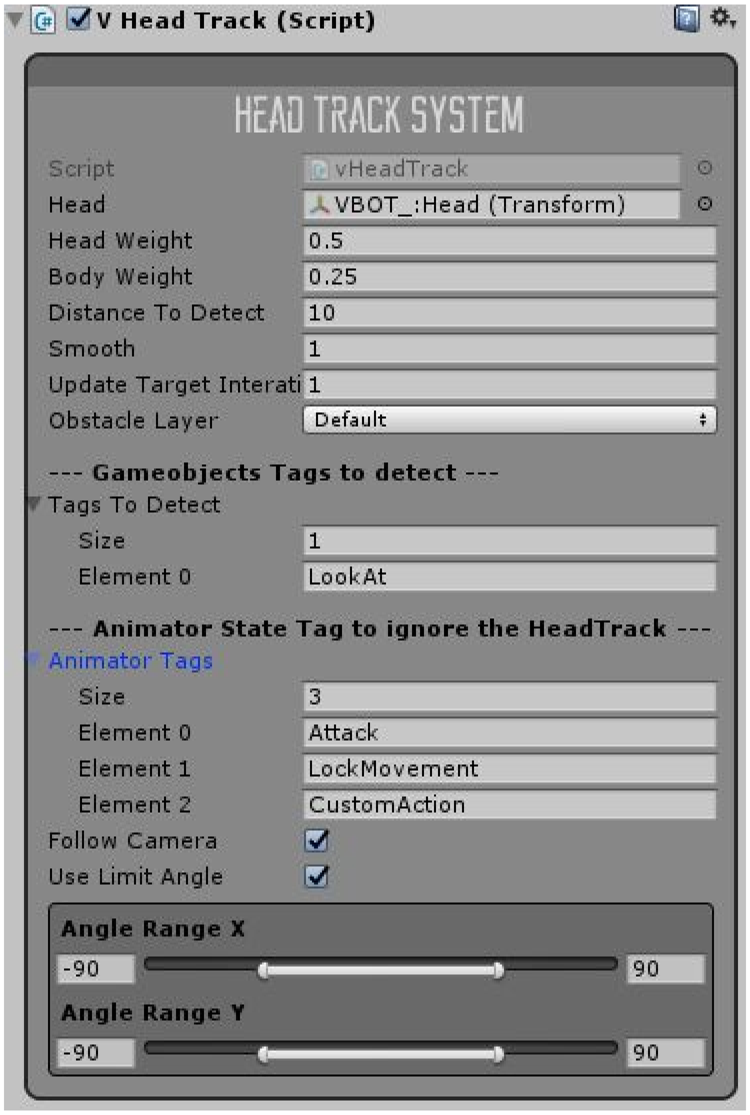
GameObject.SetActive ✓

HEAD TRACK

***Shooter** – automatically add the headtrack in order to aim up/down



Now we have a lot more options and we can use the LookAt feature as well.



V Head Track (Script)

HEAD TRACK SYSTEM

Script	vHeadTrack
Head	VBOT_:Head (Transform)
Head Weight	0.5
Body Weight	0.25
Distance To Detect	10
Smooth	1
Update Target Interati	1
Obstacle Layer	Default

--- Gameobjects Tags to detect ---

Tags To Detect	
Size	1
Element 0	LookAt

--- Animator State Tag to ignore the HeadTrack ---

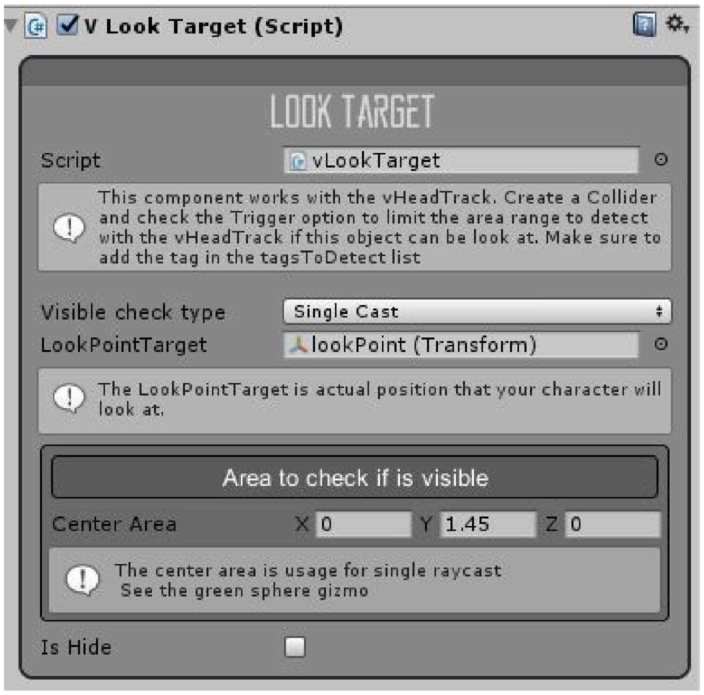
Animator Tags	
Size	3
Element 0	Attack
Element 1	LockMovement
Element 2	CustomAction
Follow Camera	✓
Use Limit Angle	✓

Angle Range X
-90 90

Angle Range Y
-90 90

If you don’t want the HeadTrack in a specific animation, you can add the Tag CustomAction into the animationState and the headtrack will turn off while this animation is playing.

To make the character look at an object, you need to add the component vLookTarget into the object, you can take a look at several examples in the DemoScenes.


V Look Target (Script)

LOOK TARGET

Script: vLookTarget

This component works with the vHeadTrack. Create a Collider and check the Trigger option to limit the area range to detect with the vHeadTrack if this object can be look at. Make sure to add the tag in the tagsToDetect list

Visible check type: Single Cast

LookPointTarget: lookPoint (Transform)

The LookPointTarget is actual position that your character will look at.

Area to check if is visible

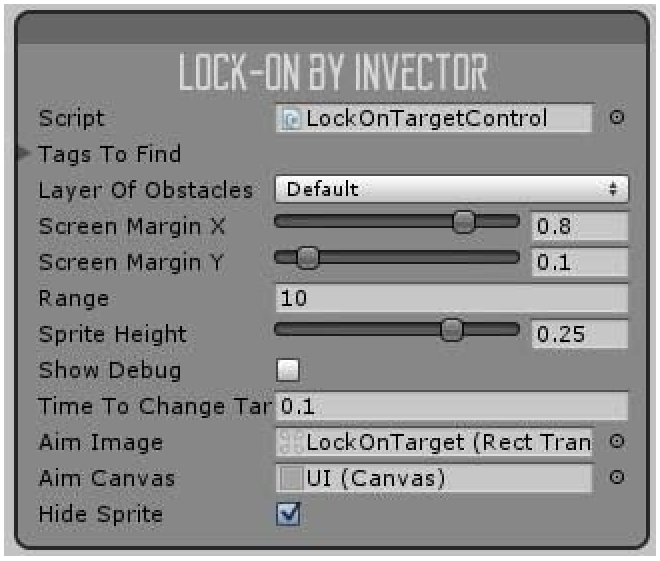
Center Area: X 0 Y 1.45 Z 0

The center area is usage for single raycast
See the green sphere gizmo

Is Hide: ☐

LOCK-ON TARGET

You can add a Lock-on component into the Camera by opening the 3rd Person Controller menu > Components > Lock-On. The component will be ready to use, you can set up the input that activate the Lock-on in the **ThirdPersonController** script, at the method **LockOnInput.**



Script	LockOnTargetControl
Tags To Find	
Layer Of Obstacles	Default
Screen Margin X	0.8
Screen Margin Y	0.1
Range	10
Sprite Height	0.25
Show Debug	☐
Time To Change Tar	0.1
Aim Image	LockOnTarget (Rect Tran
Aim Canvas	UI (Canvas)
Hide Sprite	✓

You can also display a **Sprite Image** into the Target by assigning an Image and Canvas.

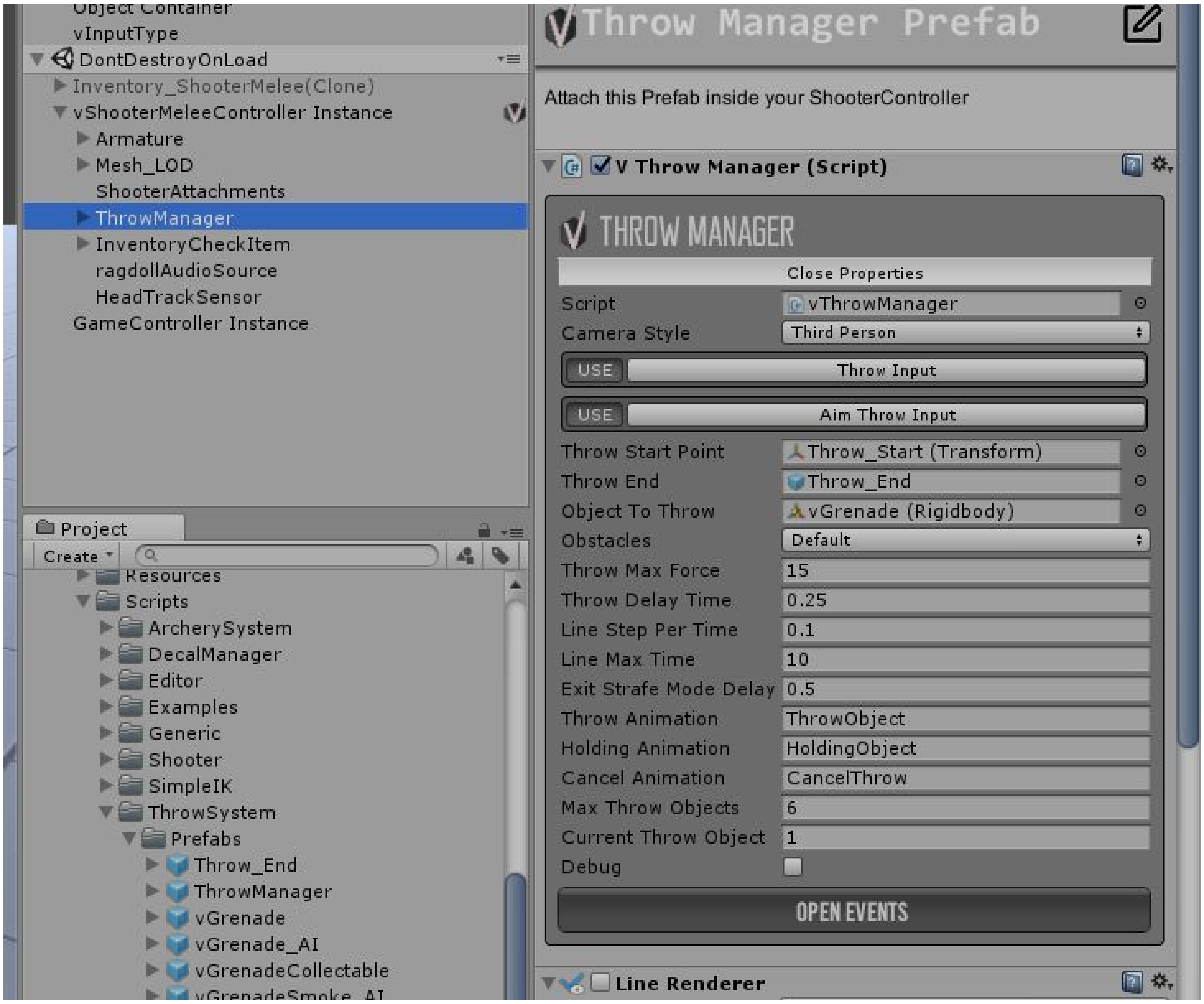
Hide Sprite will hide the sprite if the target if lock-on is false. Set off-set Y by changing the value of the **Sprite Height**.

This Lock-On currently works exclusively with our AI, it will not work out of the box with Non-Invector Characters because it needs the **vCharacter** interface to know if the target is alive. You can assign a **vCharacterStandalone** script into your gameobject, it contains health and a **TakeDamage** method to receive damage.

Shooter - You can use the Lock-On by checking the "Use Lock-On" option on the ShooterManager.

THROW OBJECT

The Throw System is pretty **Plug & Play**, just drag and drop the **ThrowManager** prefab inside your ShooterController, you can call the Event ForceToHideWeapons from the DrawHideWeapons to hide your weapons while is using the throw aiming.



Attach this Prefab inside your ShooterController

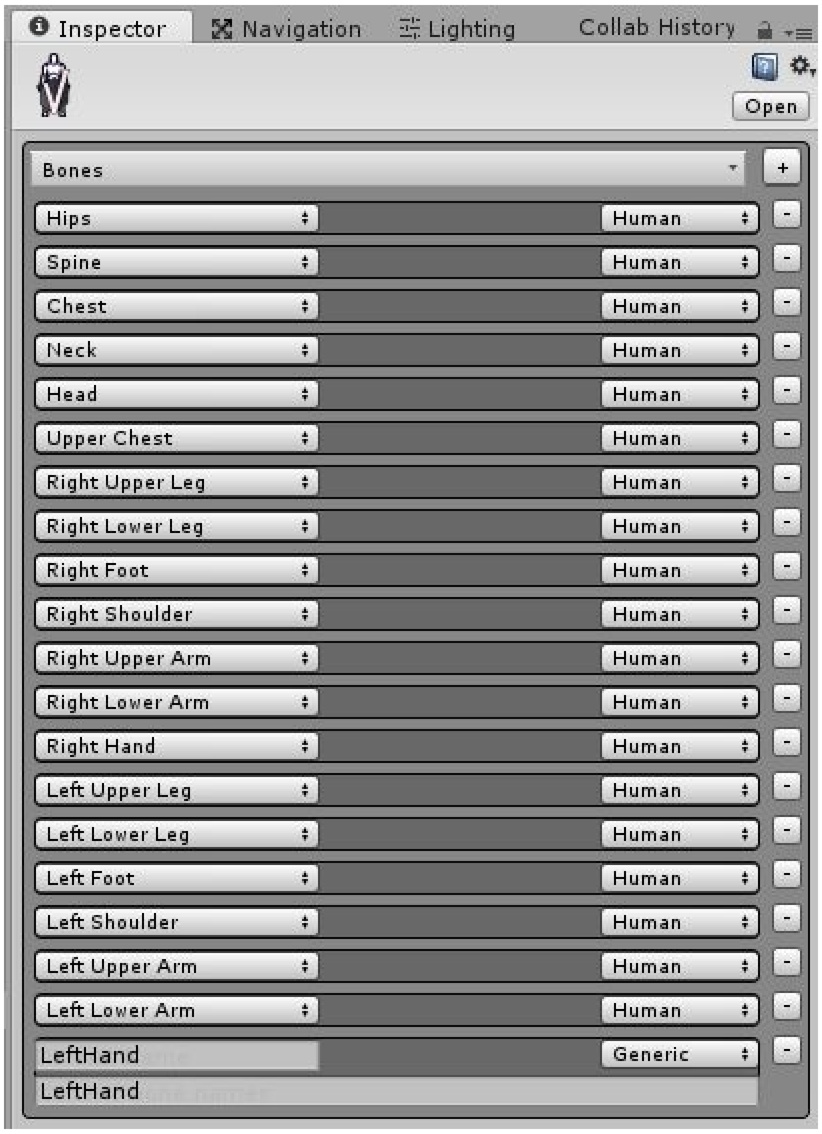
Script	vThrowManager
Camera Style	Third Person
Throw Start Point	Throw_Start (Transform)
Throw End	Throw_End
Object To Throw	vGrenade (Rigidbody)
Obstacles	Default
Throw Max Force	15
Throw Delay Time	0.25
Line Step Per Time	0.1
Line Max Time	10
Exit Strafe Mode Delay	0.5
Throw Animation	ThrowObject
Holding Animation	HoldingObject
Cancel Animation	CancelThrow
Max Throw Objects	6
Current Throw Object	1
Debug	☐

BODY SNAPPING ATTACHMENTS

We created this feature to make it easier to transfer attachments from one controller to another.

This means that you can create a Prefab of a Character Attachments and quickly add to another character, without the need of adding attachments one by one on each bone.

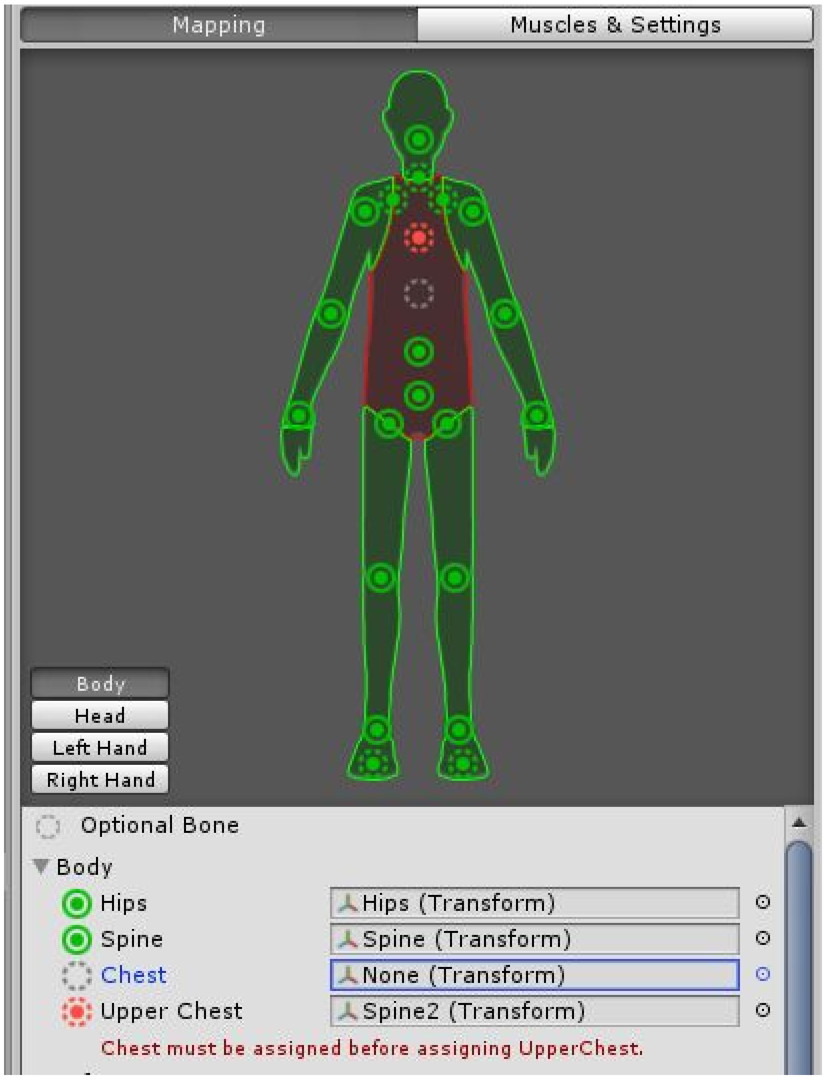
First, create an Empty GameObject inside your character, add the "**vBodySnappingControl**" and hit the "Create New BodyStruct.



Hips	Human
Spine	Human
Chest	Human
Neck	Human
Head	Human
Upper Chest	Human
Right Upper Leg	Human
Right Lower Leg	Human
Right Foot	Human
Right Shoulder	Human
Right Upper Arm	Human
Right Lower Arm	Human
Right Hand	Human
Left Upper Leg	Human
Left Lower Leg	Human
Left Foot	Human
Left Shoulder	Human
Left Upper Arm	Human
Left Lower Arm	Human
LeftHand	Generic

LeftHand

If your Avatar is already set up as Humanoid and all the bones are correctly mapped, it will all be automatically assigned for you, in some cases Unity doesn't recognize a Spine or Chest, so you need to fix by going to your Avatar and assigning the correct Bone, example:



Hips	Hips (Transform)
Spine	Spine (Transform)
Chest	None (Transform)
Upper Chest	Spine2 (Transform)

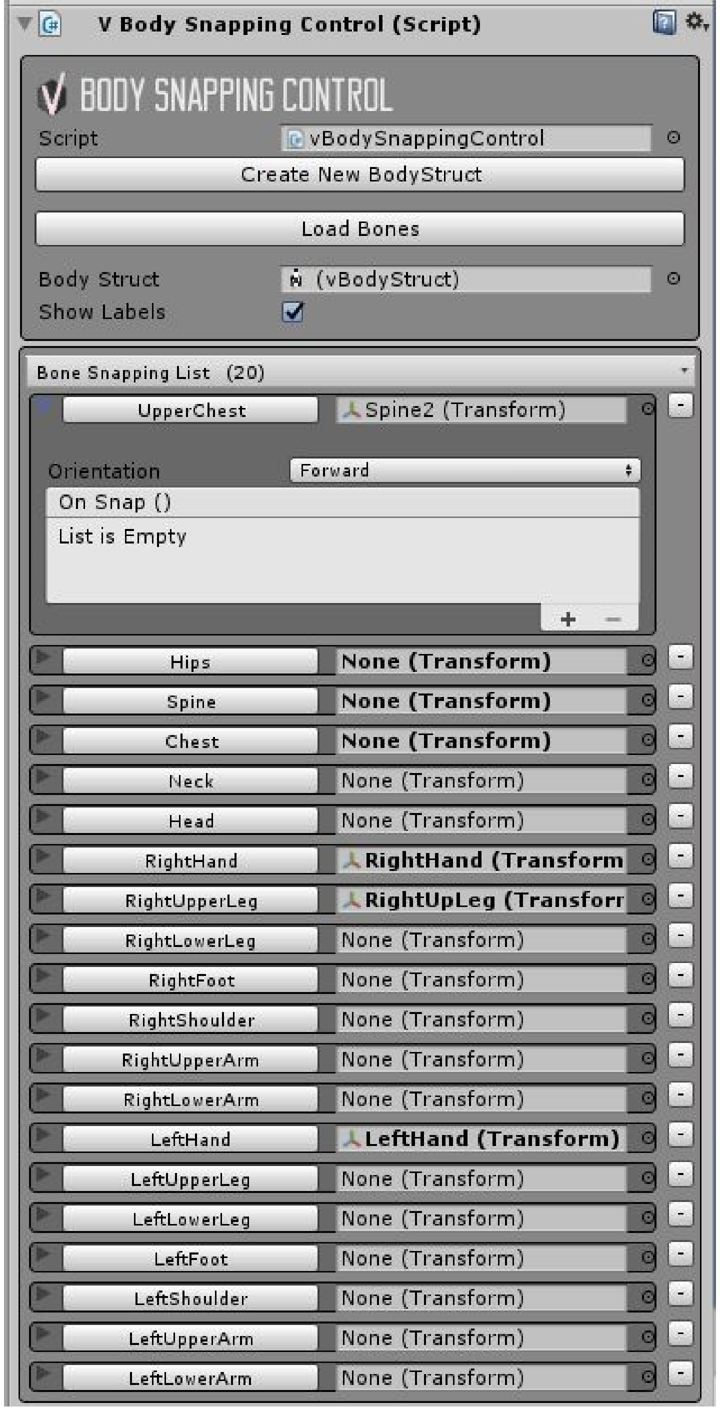
Chest must be assigned before assigning UpperChest.

Now going back to our Character BodySnap Control, you can add all your character attachments such as particles that activated on a specific bone, itemManager Handles, anything that you may use and assigned to a specific bone, once you hit Play that GameObject will be attached to the bone you assigned.

BodySnap Attachments

With this feature you can set all the attachments your character in one prefab, making it easier to transfer attachments from one character to another.

Once you hit Play the attachments will go to the desired bones.



V Body Snapping Control (Script)

BODY SNAPPING CONTROL

Script: vBodySnappingControl

Create New BodyStruct

Load Bones

Body Struct: (vBodyStruct)

Show Labels: ✓

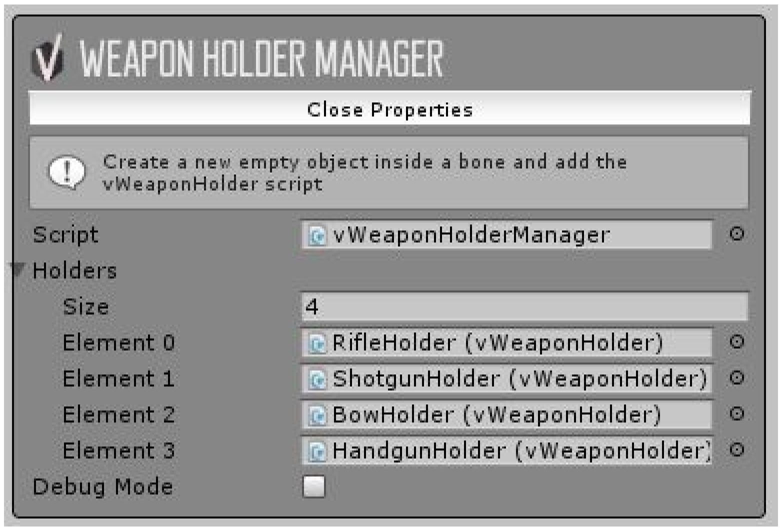
Bone Snapping List (20)

UpperChest	Spine2 (Transform)
Orientation	Forward
On Snap ()	List is Empty
Hips	None (Transform)
Spine	None (Transform)
Chest	None (Transform)
Neck	None (Transform)
Head	None (Transform)
RightHand	RightHand (Transform
RightUpperLeg	RightUpLeg (Transforr
RightLowerLeg	None (Transform)
RightFoot	None (Transform)
RightShoulder	None (Transform)
RightUpperArm	None (Transform)
RightLowerArm	None (Transform)
LeftHand	LeftHand (Transform)
LeftUpperLeg	None (Transform)
LeftLowerLeg	None (Transform)
LeftFoot	None (Transform)
LeftShoulder	None (Transform)
LeftUpperArm	None (Transform)
LeftLowerArm	None (Transform)

WEAPON HOLDER MANAGER

*This feature requires a ItemManager

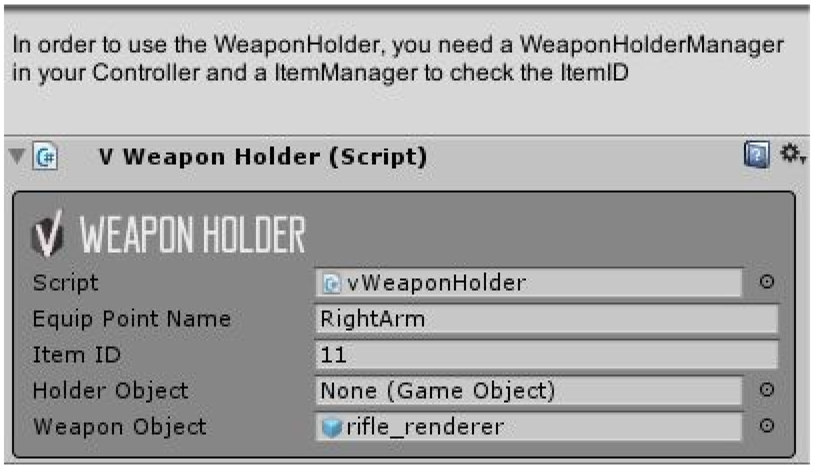
Add this component to your Shooter Controller and there is no need to setup anything here:



Create a new empty object inside a bone and add the vWeaponHolder script

```
Script          vWeaponHolderManager
Holders
  Size          4
  Element 0     RifleHolder (vWeaponHolder)
  Element 1     ShotgunHolder (vWeaponHolder)
  Element 2     BowHolder (vWeaponHolder)
  Element 3     HandgunHolder (vWeaponHolder)
Debug Mode      ☐
```

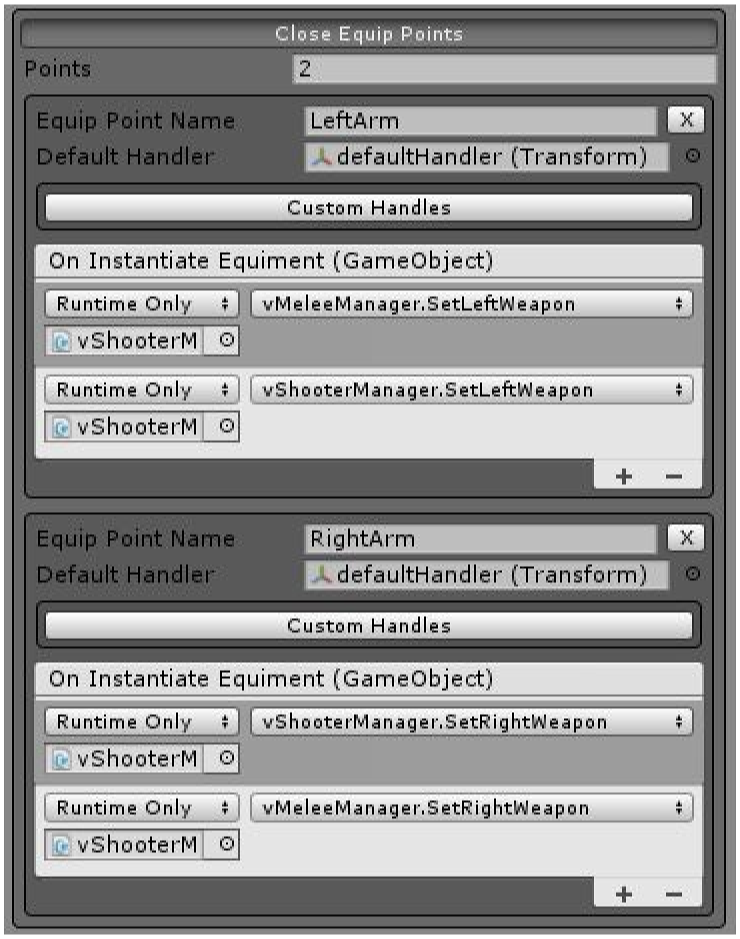
Now go inside your controller, create an empty gameObject and add the vWeaponHolder component.



In order to use the WeaponHolder, you need a WeaponHolderManager in your Controller and a ItemManager to check the ItemID

```
Script            vWeaponHolder
Equip Point Name  RightArm
Item ID           11
Holder Object     None (Game Object)
Weapon Object     rifle_renderer
```

The **EquipPointName** can be found in the **ItemManager** > EquipPoints.
By default it comes with 2 EquipPoins, LeftArm and RightArm but you also create your own custom equipPoints.



Close Equip Points

Points	2
Equip Point Name	LeftArm
Default Handler	defaultHandler (Transform)

Custom Handles

On Instantiate Equiment (GameObject)

Runtime Only	vMeleeManager.SetLeftWeapon
vShooterM	
Runtime Only	vShooterManager.SetLeftWeapon
vShooterM	

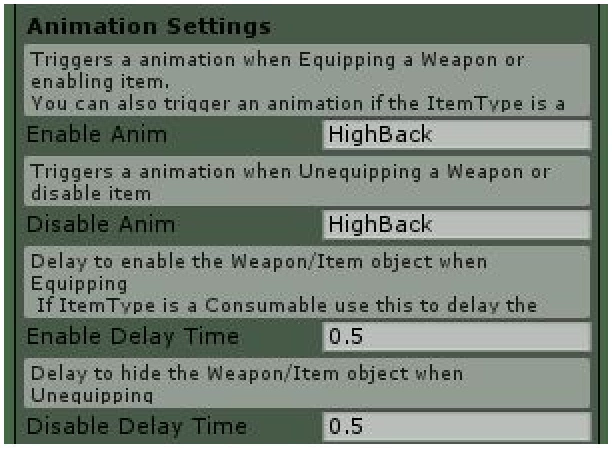
Equip Point Name	RightArm
Default Handler	defaultHandler (Transform)

Custom Handles

On Instantiate Equiment (GameObject)

Runtime Only	vShooterManager.SetRightWeapon
vShooterM	
Runtime Only	vMeleeManager.SetRightWeapon
vShooterM	

The **ItemID** can be found in the ItemManager **ItemListData**, there you can also set what animation will be played when you equip/unequip your weapon.



Animation Settings

Triggers a animation when Equipping a Weapon or enabling item.
You can also trigger an animation if the ItemType is a

Enable Anim	HighBack

Triggers a animation when Unequipping a Weapon or disable item

Disable Anim	HighBack

Delay to enable the Weapon/Item object when Equipping
If ItemType is a Consumable use this to delay the

Enable Delay Time	0.5

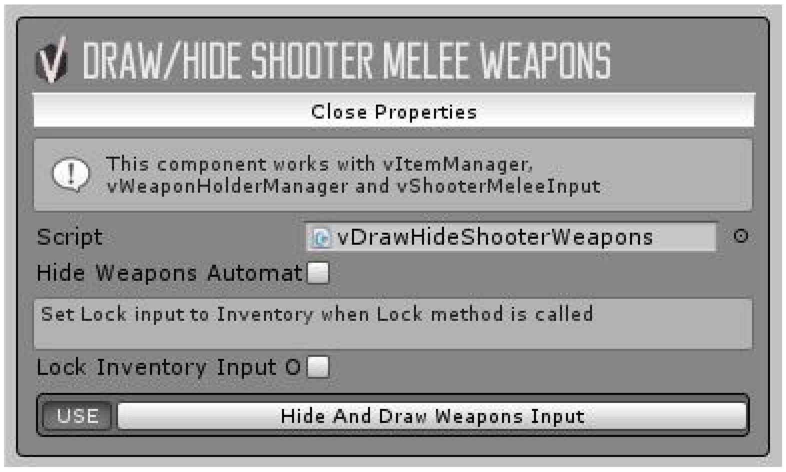
Delay to hide the Weapon/Item object when Unequipping

Disable Delay Time	0.5

The **HolderObject** is in case your weapon has a holster or something.
And the **WeaponObject** is just the 3D model without any scripts or collision, you can add all of this inside the character and leave it disabled. (Check our demo scenes to see examples)

DRAW/HIDE WEAPONS

This feature is to automatically or via input hide weapons without unequipping them.


DRAW/HIDE SHOOTER MELEE WEAPONS

Close Properties

This component works with vItemManager, vWeaponHolderManager and vShooterMeleeInput

Script vDrawHideShooterWeapons

Hide Weapons Automat

Set Lock input to Inventory when Lock method is called

Lock Inventory Input O

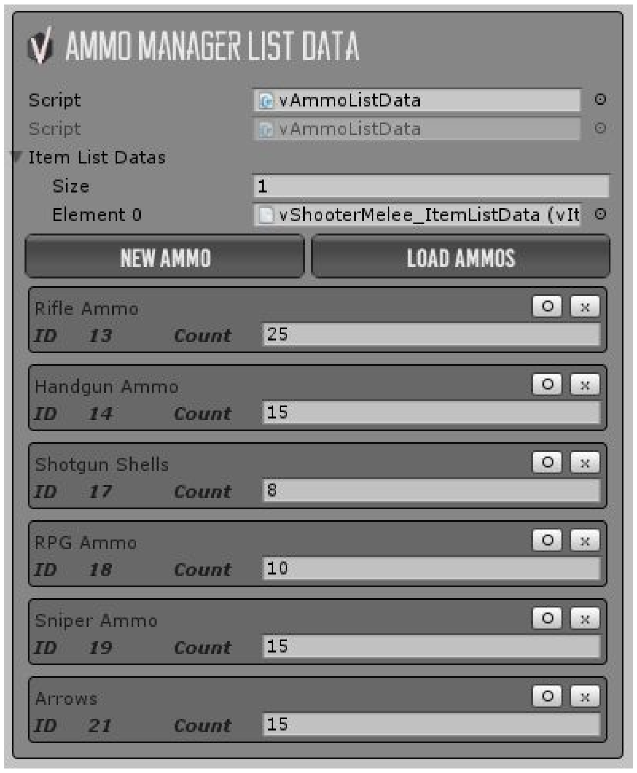
USE Hide And Draw Weapons Input

It’s pretty straightforward to use, simply add the component to your ShooterController, it must already have a ItemManager and a WeaponHolderManager already setup.

By checking the option Hide Weapons Automatically a field will appear so you can set a timer, for example after 5 seconds with the weapon equipped if will hide or you can leave it unchecked and use an Input to draw/hide the weapon.

AMMO MANAGER

To create a new AmmoListData you can duplicate the original .asset and don't forget to assign to your AmmoManager in the inspector

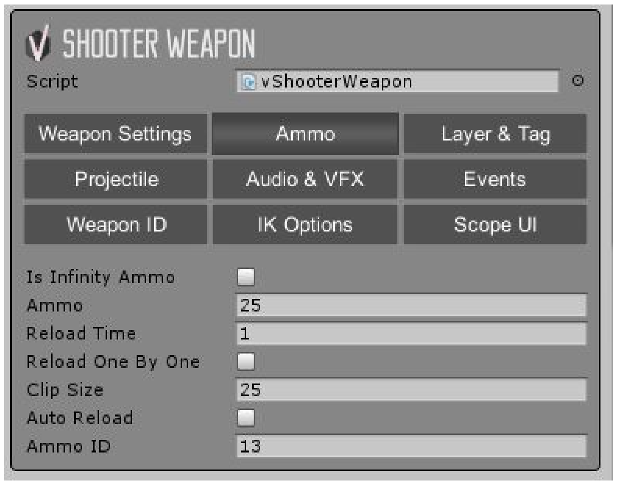


AMMO MANAGER LIST DATA

Script	vAmmoListData
Script	vAmmoListData
Item List Datas	
Size	1
Element 0	vShooterMelee_ItemListData (vIt

Name	ID	Count
Rifle Ammo	13	25
Handgun Ammo	14	15
Shotgun Shells	17	8
RPG Ammo	18	10
Sniper Ammo	19	15
Arrows	21	15

Here you can create new AmmoID for new weapons, just select your Weapon Prefab and assign the new AmmoID



SHOOTER WEAPON

Script: vShooterWeapon

Is Infinity Ammo	☐
Ammo	25
Reload Time	1
Reload One By One	☐
Clip Size	25
Auto Reload	☐
Ammo ID	13

The 'Ammo' is how much your weapon already starts with loaded, the ClipSize is how much the clip of this weapon can storage, and finally the AmmoID is the ID you created in the AmmoListData.