Third Person Engine Documentation.

Last Update: 17.07.2019.

Publisher: Renowned Studio

Author: Tamerlan Favilevich

Contents

1. Introduction

Third Person Engine is the most powerful platform on which you can easily create games from a third person. The main feature of Third Person Engine is that it has a large number of built-in functions and not sharpened certain genres is a clean engine for creating any third-person games.

Quality. Third Person Engine is constantly expanding, regular updates that improve performance, and new features are constantly added.

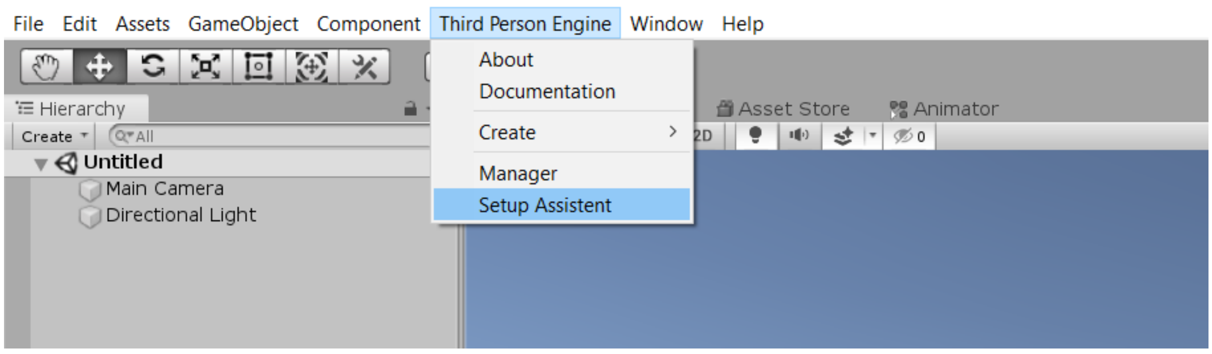
Support. We are always glad to our clients and we maintain constant contact with them, answer all questions and even help in the development of their projects, help write scripts, etc., our support is one of the best in the Asset Store.

Updates. We release only free updates, we are adherents of the principle "Once you pay and use". Absolutely all major updates and additions to the functionality will be free.

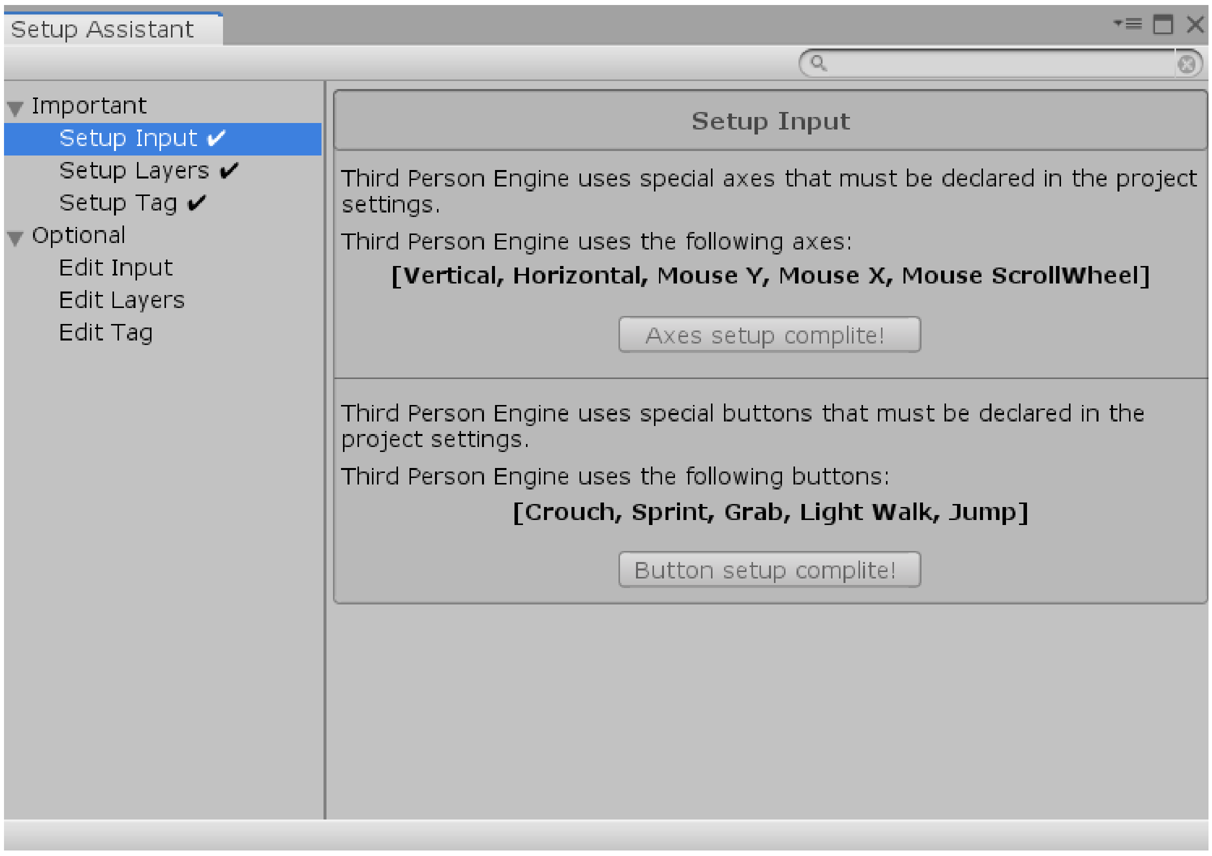
2. Start

After you download and import asset in the project go the Setup Assistant manager.

Third Person Engine → Setup Assistant



And go through all important sections. Each of them has a detailed description of what to do. After that, the project will be set up and ready to work.



Setup Input

Third Person Engine uses special axes that must be declared in the project settings.

Third Person Engine uses the following axes:

[Vertical, Horizontal, Mouse Y, Mouse X, Mouse ScrollWheel]

Axes setup complite!

Third Person Engine uses special buttons that must be declared in the project settings.

Third Person Engine uses the following buttons:

[Crouch, Sprint, Grab, Light Walk, Jump]

Button setup complite!

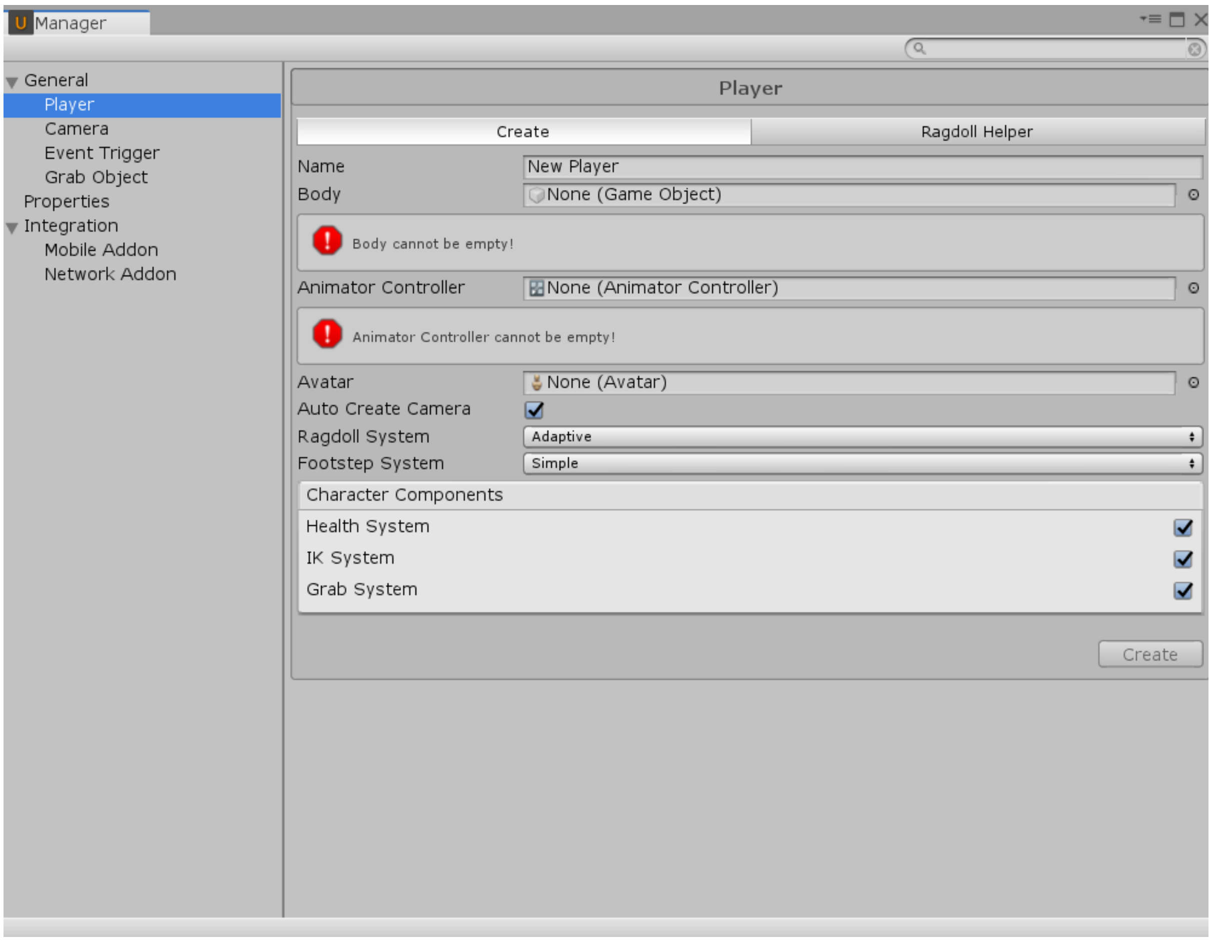
3. Player

3.1. Create Player

Drag and drop you character model on scene, remove all components from you model and create for it ragdoll by default Unity Ragdoll Wizard.

Open Manager and select “Player” section add fill all required fields.

1. ***Body:*** Your ragdolled player gameobject.
2. ***Animator Controller:*** Base animator build-in in asset.
3. ***Avatar:*** Your model avatar.



Name: New Player
Body: None (Game Object)

Body cannot be empty!

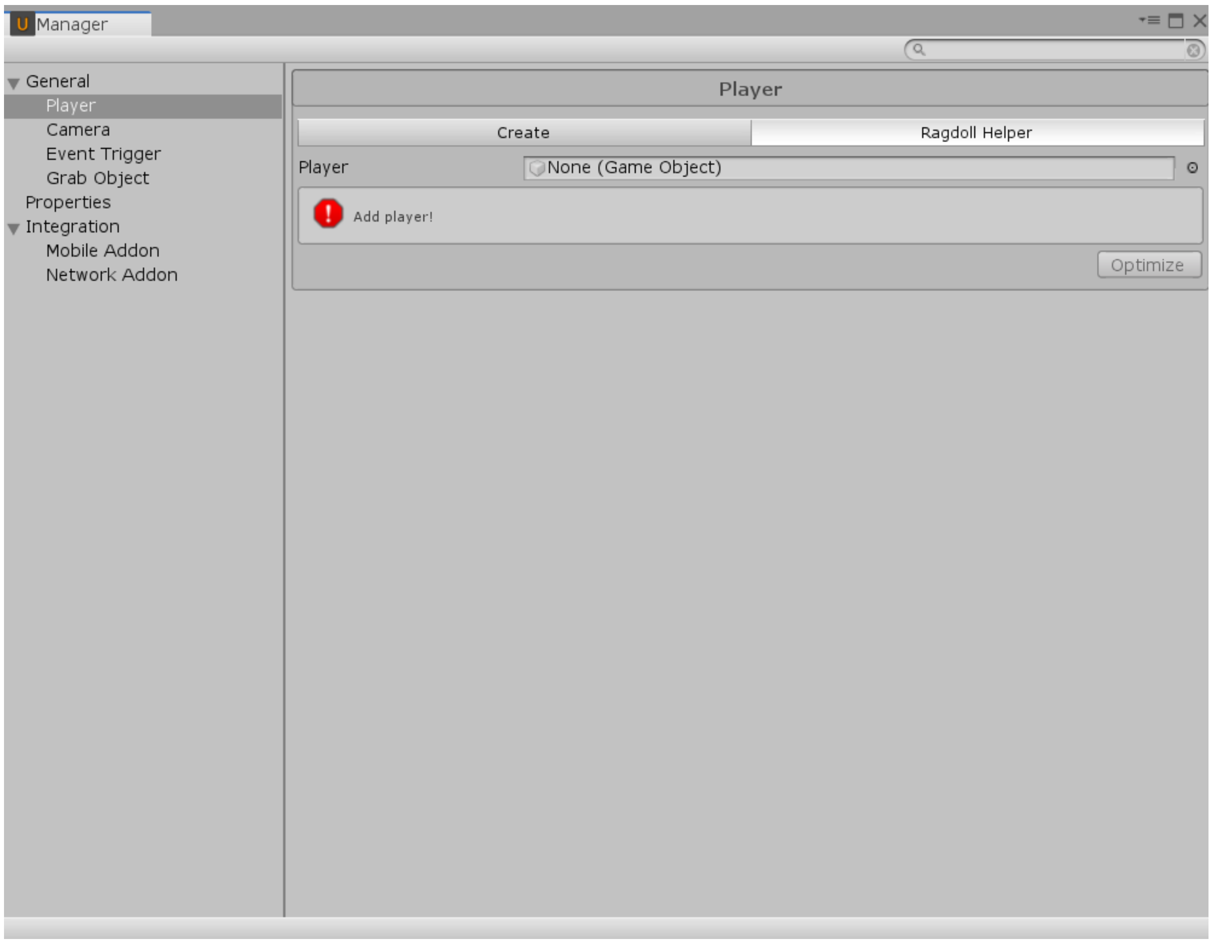
Animator Controller: None (Animator Controller)

Animator Controller cannot be empty!

Avatar: None (Avatar)
Auto Create Camera: ✓
Ragdoll System: Adaptive
Footstep System: Simple

Character Components
Health System ✓
IK System ✓
Grab System ✓

After you player successfully created open Radoll Helper add your player and press “Optimize” button.



Player None (Game Object)

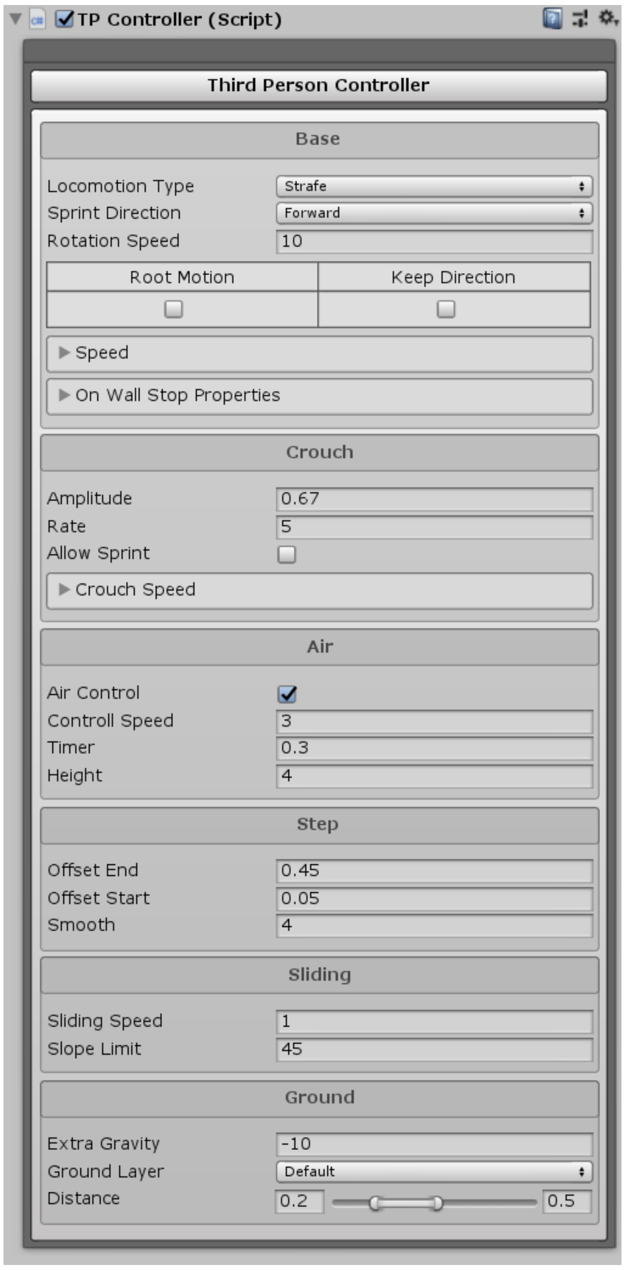
Add player!

After these steps, the player is completely ready and configured for the game!

3.2 Player Components

3.2.1. TPController Component

Base component of the player handles the basic functions of movement. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.
Requirement: [Required]



TP Controller (Script)

Third Person Controller

Base

Locomotion Type	Strafe
Sprint Direction	Forward
Rotation Speed	10

Root Motion	Keep Direction
☐	☐

▶ Speed

▶ On Wall Stop Properties

Crouch

Amplitude	0.67
Rate	5
Allow Sprint	☐

▶ Crouch Speed

Air

Air Control	✓
Controll Speed	3
Timer	0.3
Height	4

Step

Offset End	0.45
Offset Start	0.05
Smooth	4

Sliding

Sliding Speed	1
Slope Limit	45

Ground

Extra Gravity	-10
Ground Layer	Default
Distance	0.2 – 0.5

Fields Description

1. **[Base]**
   1.1. ***Locomotion Type:*** Controller locomotion type.
   1.2. ***Sprint Direction:*** Controller sprint direction conditions.
   1.3. ***Rotation Speed:*** Character rotation speed.
   1.4. ***Root Motion:*** Root motion for character.
   1.5. ***Keep Direction:*** Keep character direction by camera direction.
   1.6. [Speed Properties]
      1.6.1. ***Walk:*** Walk speed.
      1.6.2. ***Run:*** Run speed.
      1.6.3. ***Sprint:*** Sprint speed.
   1.7. [On Wall Stop Properties]
      1.7.1. ***Check Range:*** At what range from the character to check the presence of the wall.
      1.7.2. ***Angle:*** At what angle relative to the wall the character can't move.
      1.7.3. ***Wall Layer:*** Wall layer mask.
      1.7.4. ***Check Conditions:*** Check on wall stop conditions. True: System is active. False: System not active.
2. **[Crouch Properties]**
   2.1. ***Amplitude:*** Crouching amplitude. Formula:(CH = H-A). Where: CH -Crouch character height, H-Character height, A-Crouch amplitude. Example: 1.8(H) - 0.67(A) = 1.13 (CH).
   2.2. ***Rate:*** The rate of change in the height of the Collider.
   2.3. ***Allow Sprint:*** Allow character sprint while crouching.
3. **[Air Properties]**
   3.1. ***Air Control:*** Allow control character while in air.
   3.2. ***Timer:*** How long will be active jumping state after jumping.
   3.3. ***Forward Impulse:*** Character forward jump impulse.
   3.4. ***Control Speed:*** Air control speed.
   3.5. ***Height:*** Jump height.
4. **[Step Properties]**
   4.1. ***Offset End:*** Character step offset end.

4.2. ***Offset Start:*** Character step offset start.

4.3. ***Smooth:*** Character step smooth.

5. [Sliding Properties]

5.1. ***Slope Limit*****:** Slope limit for start sliding.

5.2. ***Sliding Speed*****:** Character speed while sliding.

6. [Ground Properties]

6.1. ***Ground Layer*****:** Layer for detect ground.

6.2. ***Distance*****:** Ground min/max distance.

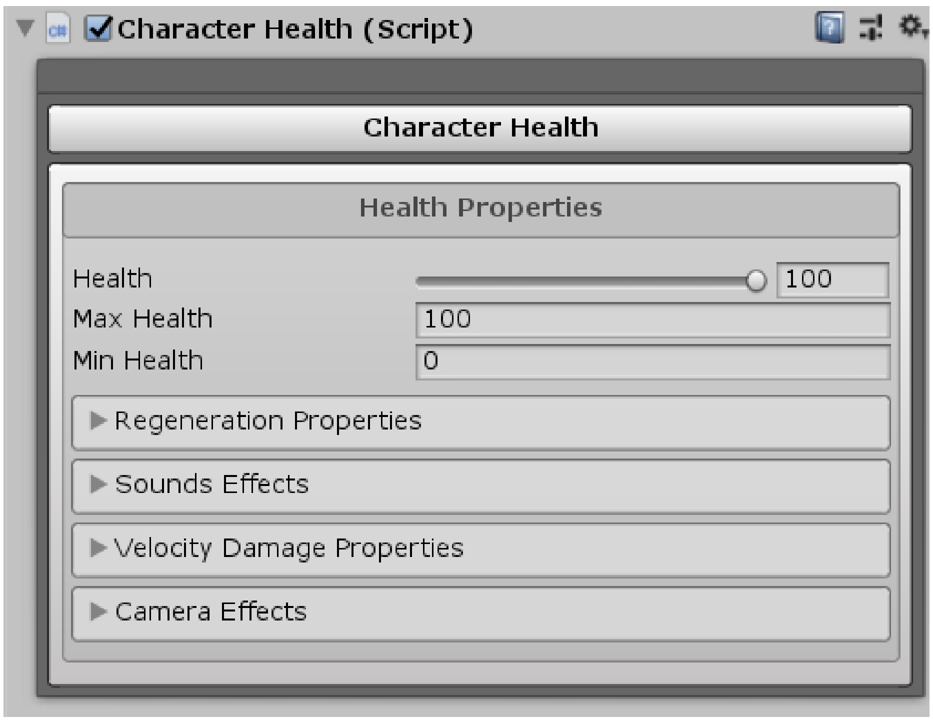
6.3. ***Min Limit:*** Ground min distance.

6.4. ***Max Limit*****:** Ground max distance.

3.2.2. Character Health Component.

Base health system, handle health properties of the player. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.

Requirement: [Optional]



Character Health (Script)

Character Health

Health Properties

Health 100

Max Health 100

Min Health 0

▶ Regeneration Properties

▶ Sounds Effects

▶ Velocity Damage Properties

▶ Camera Effects

Fields Description

1. ***[Health Properties]***

   1.1. ***Health***: Health point value.

   1.2. ***Max Health***: Max health point value.

   1.3. ***Min Health***: Min health point value.

   1.4. [Regeneration Properties]

      1.4.1. ***Rate***: Rate (in seconds) of adding health points. (V/R - Value per rate).

      1.4.2. ***Value***: Health point value.

      1.4.3. ***Delay***: Delay before start adding health.

   1.5. Sounds Effects Properties

      1.5.1. ***Take Damage***: Sound will be player when player take damage.

1.5.2. ***Velocity Damage*:** Sound will be played when player take damage from velocity speed.

1.5.3. ***Heartbeat*:** Heartbeat sound will player every rate, if condition [Heartbeat Start From] is met.

1.5.4. ***Heartbeat Rate*:** Heartbeat sound play rate.

1.5.5. ***Heartbeat Start From*:** Start play heartbeat sound if player health <= this value.

1.5.6. ***Death*:** Death sound will player when player die.

1.6. [Camera Effects Properties]

1.6.1. ***Profile*:** Post processing profile assets.

1.6.2. ***Start From*:** From how many percent of health begin to show the effect.

1.6.3. ***Reset Speed*:** Speed for resetting effect, when health more then start point value.

1.6.4. ***CA Speed*:** Chromatic aberration effect speed.

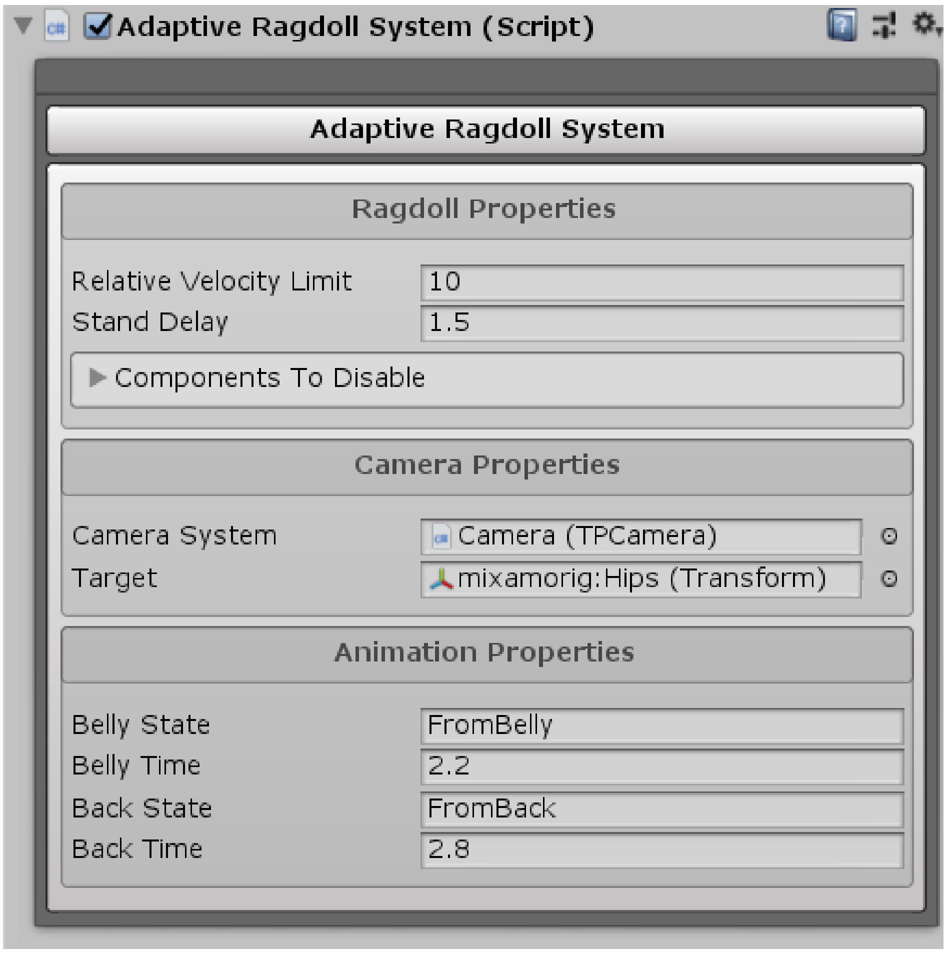
1.6.5. ***Vignette Smooth:*** Vignette effect smooth value.

1.6.6. ***Vignette Range***: Vignette intensity range effect.

3.2.3. Adaptive Ragdoll System

Used for automatically handle ragdoll based on physical impact on the character. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.

Requirement: [Optional]



Adaptive Ragdoll System (Script)

Adaptive Ragdoll System

Ragdoll Properties

Relative Velocity Limit	10
Stand Delay	1.5

▶ Components To Disable

Camera Properties

Camera System	Camera (TPCamera)
Target	mixamorig:Hips (Transform)

Animation Properties

Belly State	FromBelly
Belly Time	2.2
Back State	FromBack
Back Time	2.8

Fields Description

1. **[Ragdoll Properties]**
   - 1.1. ***Relative Velocity Limit***: Limits of the relative linear velocity of the two colliding objects for the character to start ragdoll system.
   - 1.2. ***Stand Delay***: Delay before character stand. after ragdoll is played and wait in stable position.
   - 1.3. ***Components To Disable***: Components array which will disable while ragdoll is active.
2. **[Camera Properties]**

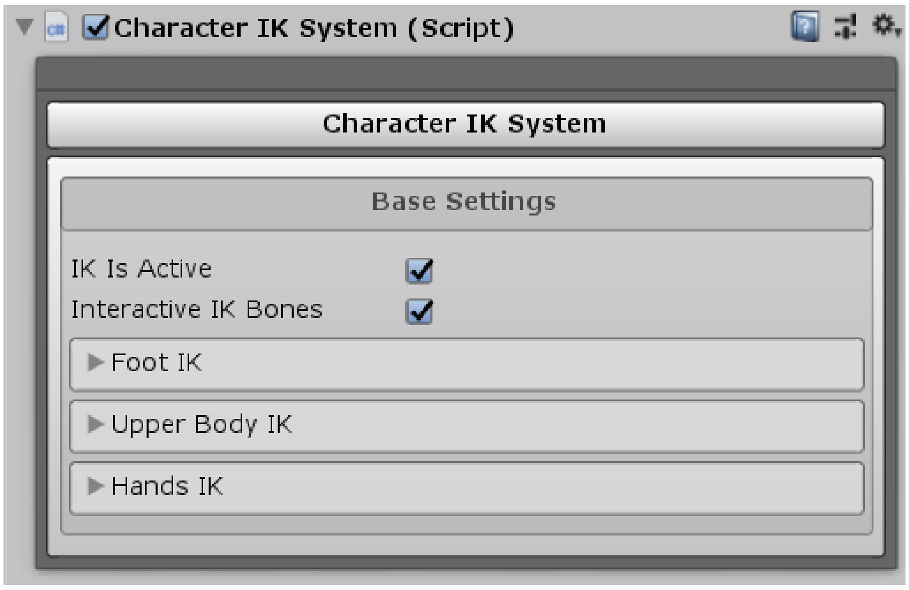
2.1. ***Camera System*:** Camera system instance.
2.2. ***Target*:** Target while ragdoll is active.

3. [Animation Properties]

3.1. ***Belly State*:** Animator state name to get up from belly.
3.2. ***Back State*:** Animator state name to get up from back.
3.3. ***Belly Time*:** Belly animation time.
3.4. ***Back Time*:** Back animation time.

3.2.4. Character IK System

Base Inverse Kinematics system for third person engine. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.
Requirement: [Optional]


Character IK System (Script)
Character IK System
Base Settings
IK Is Active ✓
Interactive IK Bones ✓
▶ Foot IK
▶ Upper Body IK
▶ Hands IK

Fields Description

1. **[Base Settings]**
   - 1.1. ***IK Is Active:*** IK active state.
   - 1.2. ***Interactive IK Bones:*** (Only in Editor) Display bones and IK interactively at the scene window.
   - **1.3. [Foot IK]**
     - 1.3.1. ***Left Foot:*** Left foot bone.
     - 1.3.2. ***Right Foot:*** Right foot bone.
     - 1.3.3. ***Ground Layer:*** Layer to detect ground by foot ray.
     - 1.3.4. ***Foot Offset:*** Foot offset distance between ground and foot.
     - 1.3.5. ***Delta Amplifier:*** Delta amplifier for character weight.
     - 1.3.6. ***Collider Smooth:*** Collider height change speed.
     - 1.3.7. ***Foot Rotation:*** Process foot rotation by surface normal.
   - **1.4. [Upper Body IK]**
     - 1.4.1. ***Look Target:*** Look target for character height.
     - 1.4.2. ***Weight:*** Upper body IK weight.
     - 1.4.3. ***Body Weight:*** Body IK weight.
     - 1.4.4. ***Head Weight:*** Head IK weight.
     - 1.4.5. ***Eyes Weight:*** Eyes IK weight.
     - 1.4.6. ***Clamp Weight:*** Clamp IK weight.
   - **1.5. [Hands IK]**
     - 1.5.1. ***Left Hand Target:*** Left hand target transform.
     - 1.5.2. ***Right Hand Target:*** Right hand target transform.
     - 1.5.3. ***Smooth:*** Hand IK smooth.

3.2.5. Footstep System

Third Person Engine contains two footsteps systems, Simple Footstep System and Advanced Footstep System.

Simple Footstep System - Plays the step sound cyclically after player moved certain interval.

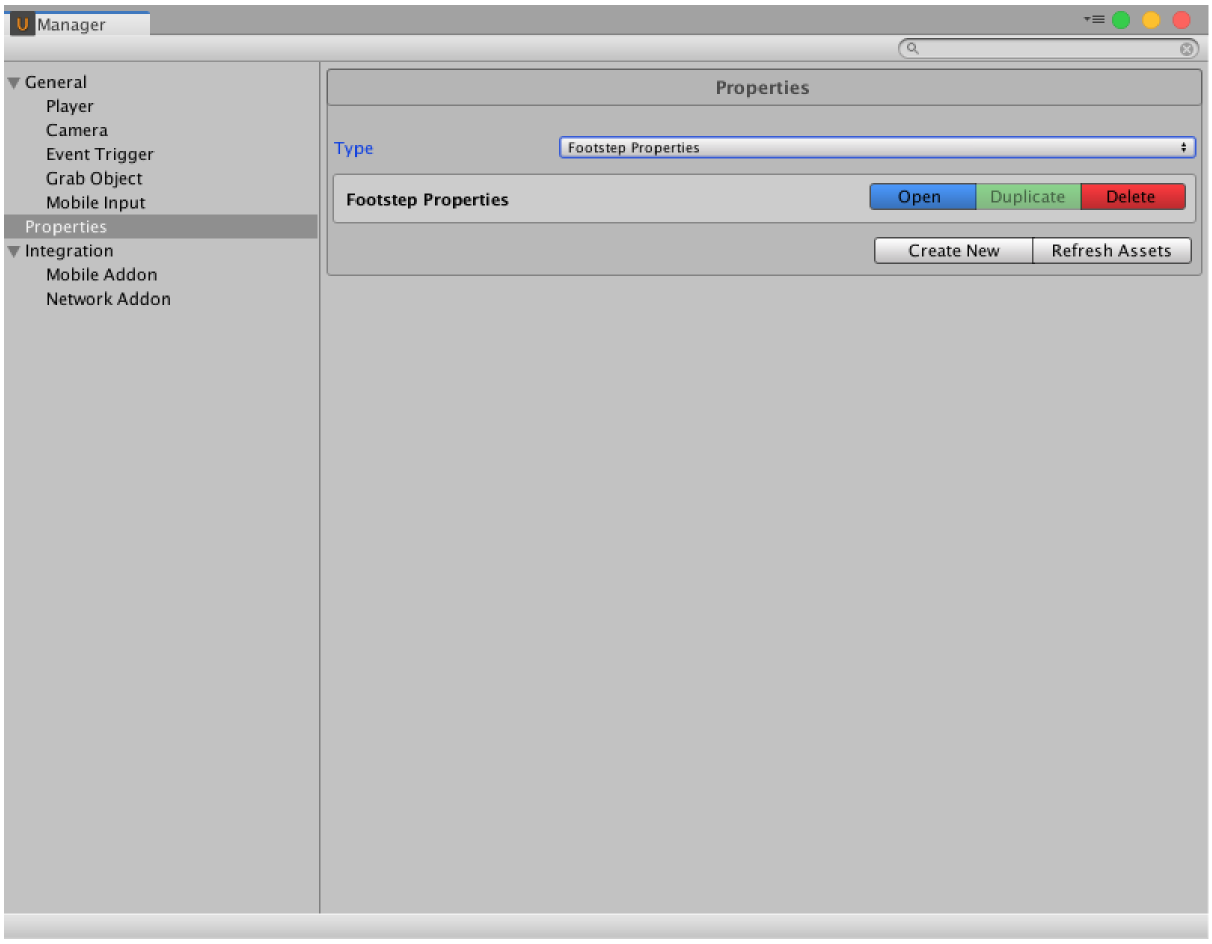
Advanced Footstep System - Extends of Simple Footstep Sound, and unlike it, the system processes the pitch sound for each foot separately, that is, if the character runs between two different surfaces, the sounds of these surfaces will be played alternately.

Both system use Footstep property asset it’s asset containing mapping of (texture or physics material) to (step, jump and land) sounds.

Requirement: [Optional]

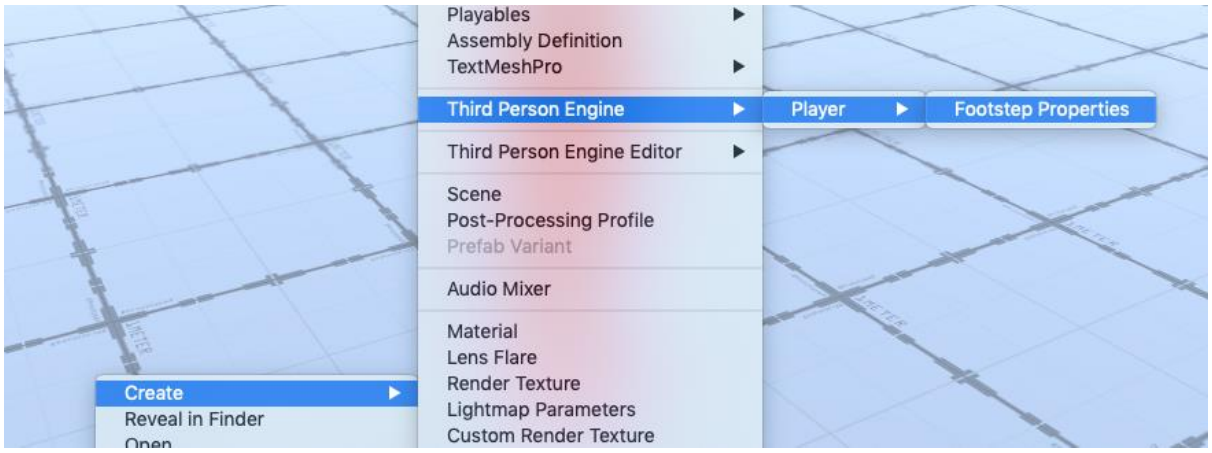
3.2.5.1. Create Footstep Property

For create footstep property go to Manager, section Properties and select type Footstep Properties. After will be displayed all Footsteps Properties in the project. You can create a new one or duplicate an existing one and change it.

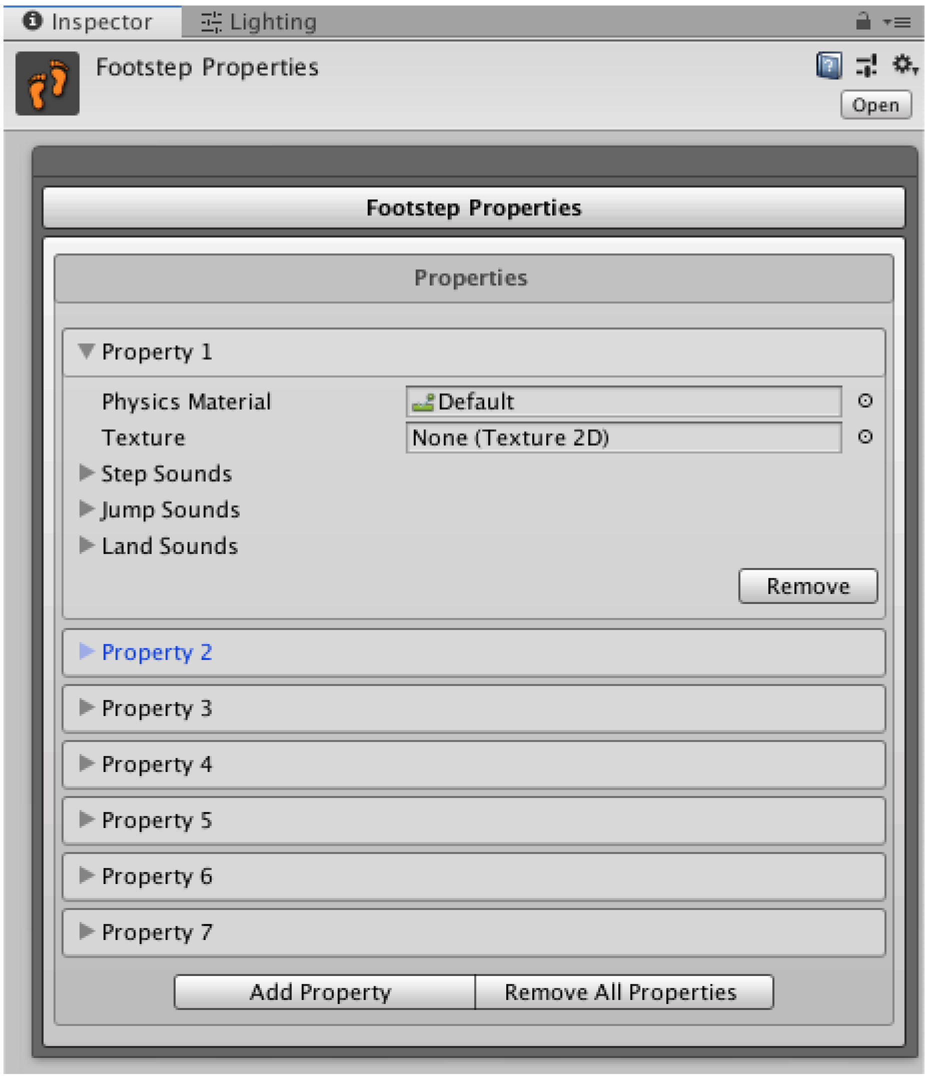


Another quick way to create an asset is through the drop-down menu. Press right click on project window and select:

Create/Third Person Engine/Player/Footstep Properties



3.2.5.2. Footstep Property



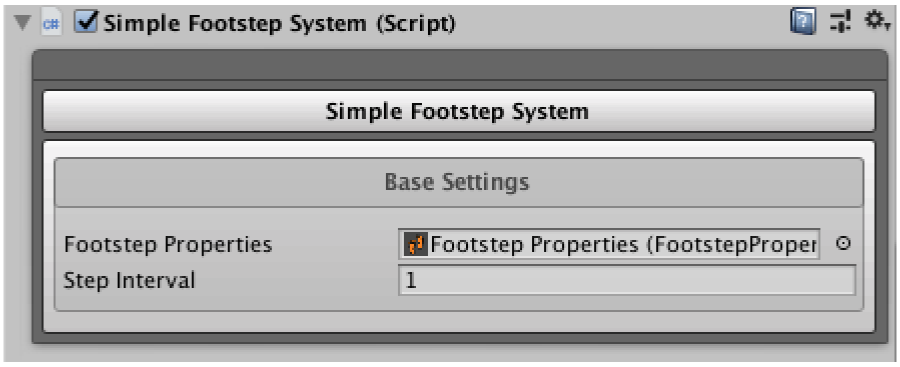
For create new property press “Add Property” button, for remove all properties containing in current asset press “Remove All Properties” and for remove separately property, expand property and press “Remove”.

Fields Description

1. **[Properties]**
   1.1. ***Physics Material:*** Physics Material of surface.
   1.2. ***Texture:*** Texture of surface.
   1.3. ***Jump Sounds:*** Jump sounds array (player randomly).
   1.4. ***Step Sounds:*** Step sounds array (player randomly).
   1.5. ***Land Sounds:*** Land sounds array (player randomly).

3.2.5.2 Simple Footstep System

Simple Footstep System - Plays the step sound by footstep properties cyclically after player moved step interval. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.


Simple Footstep System (Script)

Simple Footstep System

Base Settings

Footstep Properties: Footstep Properties (FootstepProper

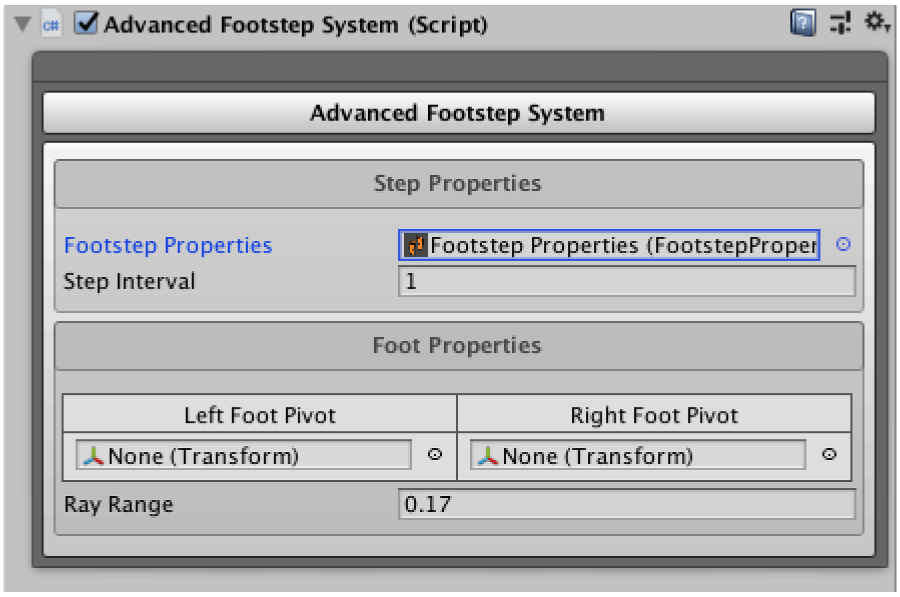
Step Interval: 1

Fields Description

1. **[Base Settings]**
   1.1. ***Footstep Properties:*** Footstep Properties asset.
   1.2. ***Step Interval:*** After each step will be played footstep sound.

3.2.5.2. Advanced Footstep System

Advanced Footstep System - Extends of Simple Footstep Sound, and unlike it, the system processes the pitch sound for each foot separately, that is, if the character runs between two different surfaces, the sounds of these surfaces will be played alternately. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.



Advanced Footstep System (Script)

Advanced Footstep System

Step Properties

Footstep Properties: Footstep Properties (FootstepProper

Step Interval: 1

Foot Properties

Left Foot Pivot	Right Foot Pivot
None (Transform)	None (Transform)

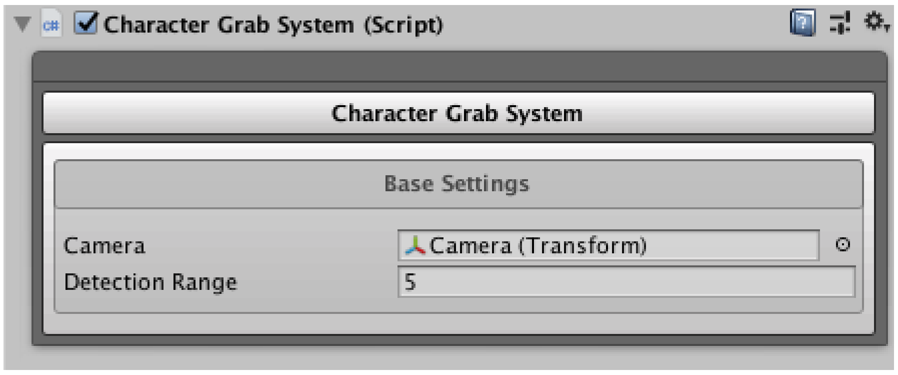
Ray Range: 0.17

Fields Description

1. **[Step Settings]**
   1.1. ***Footstep Properties:*** Footstep Properties asset.
   1.2. ***Step Interval:*** After each step will be played footstep sound.
2. **[Foot Properties]**
   2.1. ***Left Foot Pivot:*** Character left foot pivot.
   2.2. ***Right Foot Pivot:*** Character right foot pivot.
   2.3. ***Ray Range:*** Ray range from foot to the ground.

3.2.6. Character Grab System

Base character grabbing system. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.
Requirement: [Optional]



Character Grab System (Script)

Character Grab System

Base Settings

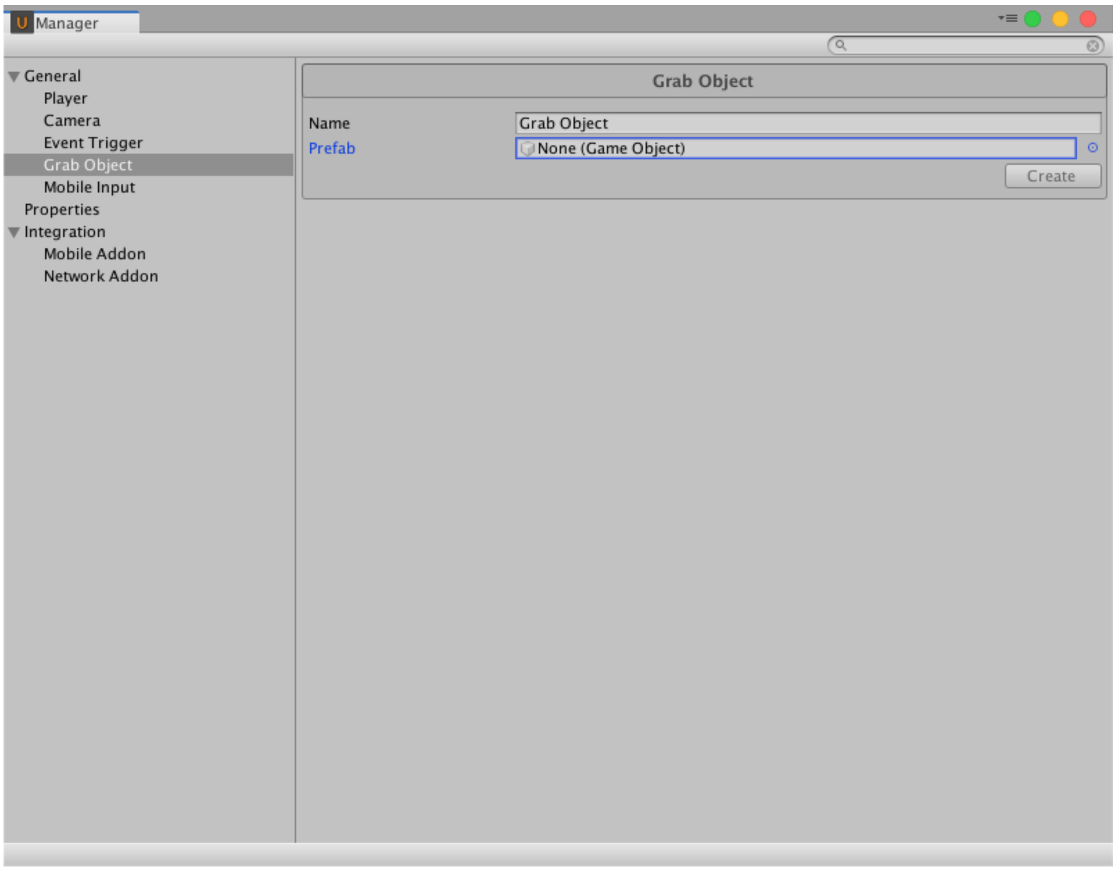
Camera: Camera (Transform)
Detection Range: 5

Fields Description

1. **[Base Settings]**
   1.1. ***Camera:*** Character camera.
   1.2. ***Detection Range:*** Ray range from camera.

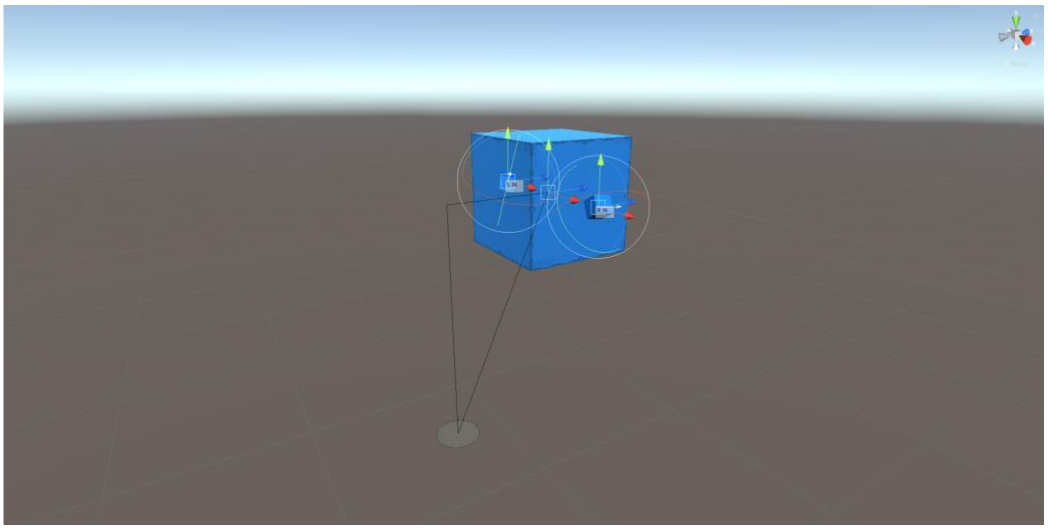
3.2.6.1. Create Grab Object

For create grab object go to Manager, section Grab Object. Add the object you want to make grabble, optionally, set specify a name and press “Create”. After this grab object will be created.



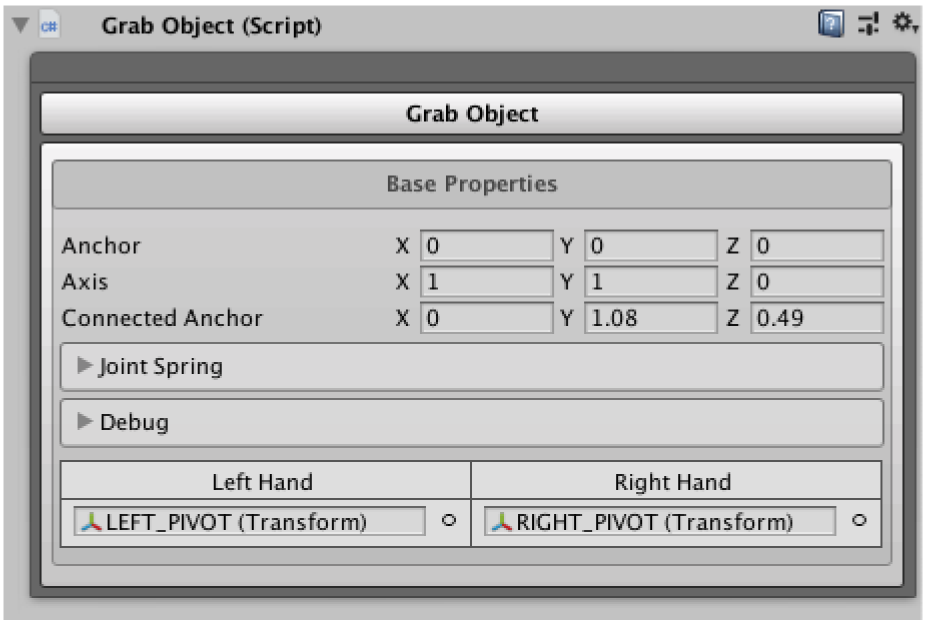
Name: Grab Object
Prefab: None (Game Object)

↓



3.2.6.2 Grab Object

Base component for detect current object. In other words, this component marks the object like a grabble and wraps it with the necessary properties. All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.


Grab Object (Script)

Grab Object

Base Properties

	X	Y	Z
Anchor	0	0	0
Axis	1	1	0
Connected Anchor	0	1.08	0.49

▶ Joint Spring

▶ Debug

Left Hand	Right Hand
LEFT_PIVOT (Transform)	RIGHT_PIVOT (Transform)

Fields Description

1. **[Base Properties]**
   - 1.1. ***Anchor:*** The position of the axis around which the body swings. The position is defined in local space.
   - 1.2. ***Axis:*** The direction of the axis around which the body swings. The direction is defined in local space.
   - 1.3. ***Connected Anchor:*** Optional reference to the Rigidbody that the joint is dependent upon. If not set, the joint connects to the world.
   - **1.4. [Joint Spring]**
     - 1.4.1. ***Spring:*** The force the object asserts to move into the position.
     - 1.4.2. ***Damper:*** The higher this value, the more the object will slow down.
     - 1.4.3. ***Target Position:*** Target angle of the spring. The spring pulls towards this angle measured in degrees.
   - **1.5. [Debug]**

1.5.1. ***Connected Anchor GUI:*** Draw connected anchor GUI.

1.5.2. ***Hands IK GUI:*** Draw hands IK GUI.

1.6. ***Left Hand:*** Left Hand IK Target.

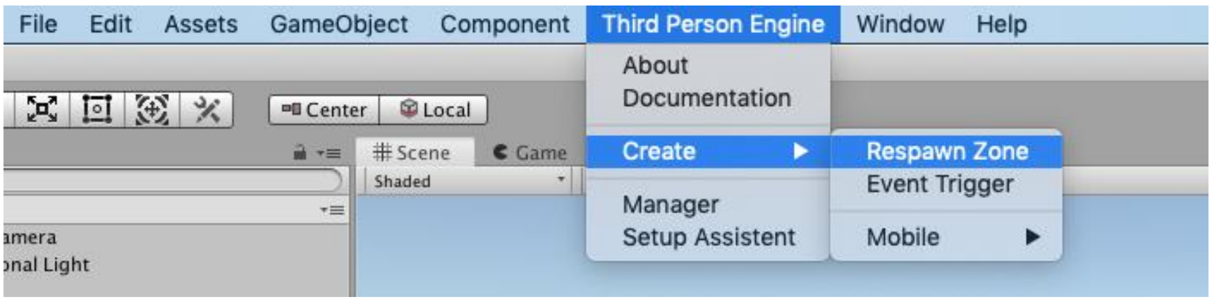
1.7. ***Right Hand:*** Right Hand IK Target.

4. Respawn Zone

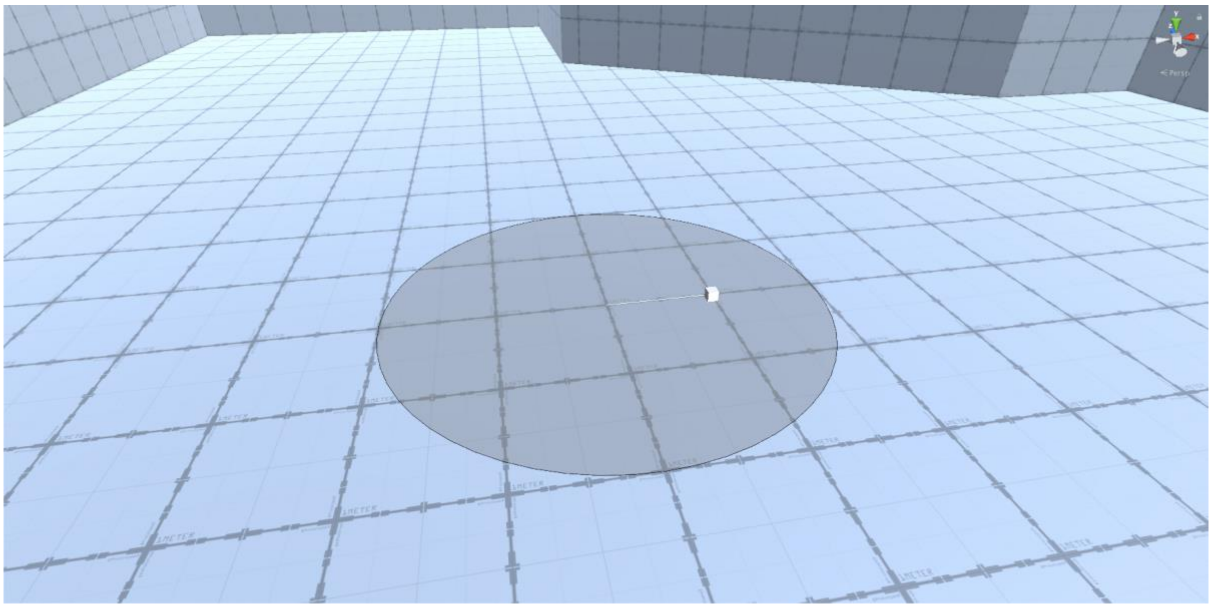
4.1 Create Respawn Zone

For create new respawn manager go to the MenuItem. After this respawn zone will be created at the scene.

Third Person Engine/Create/Respawn Zone

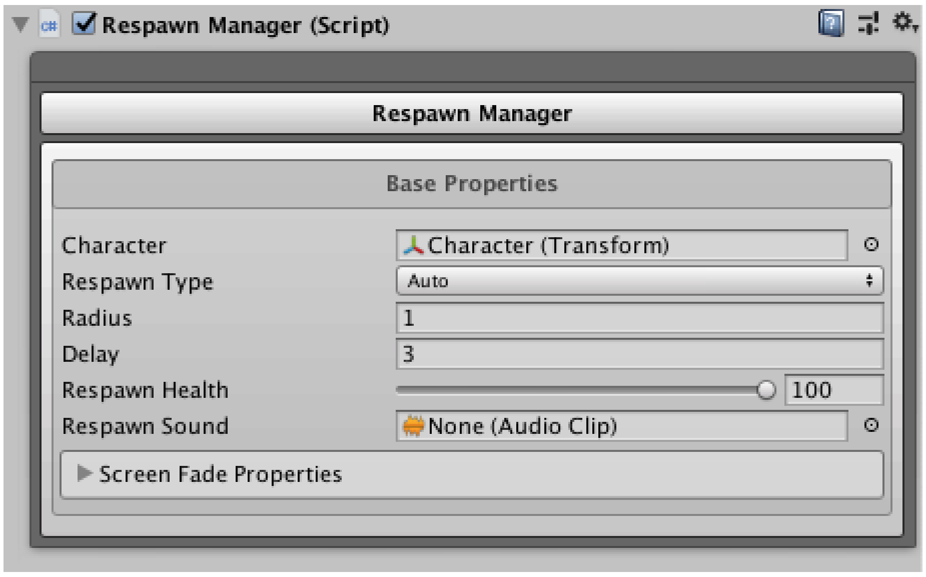


↓



4.2 Respawn Manager

All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.


Respawn Manager (Script)

Respawn Manager

Base Properties

Character	Character (Transform)
Respawn Type	Auto
Radius	1
Delay	3
Respawn Health	100
Respawn Sound	None (Audio Clip)

▶ Screen Fade Properties

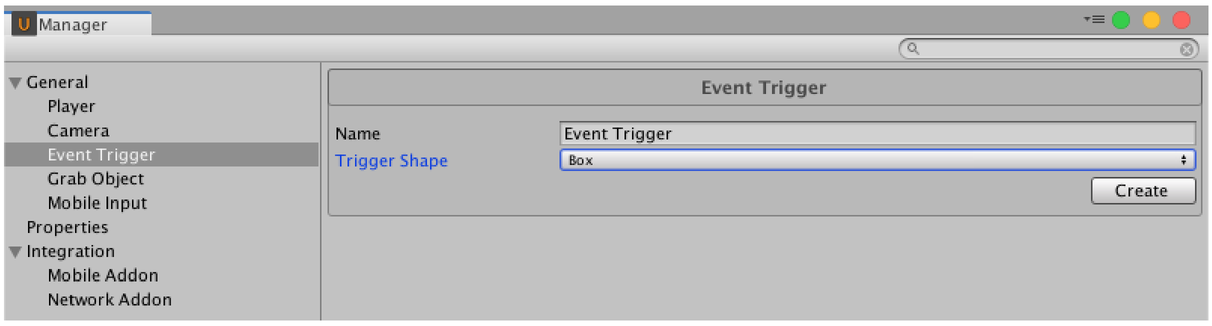
Fields Description

1. **[Base Properties]**
   1.1. ***Character:*** Character transform.
   1.2. ***Respawn Type:*** Auto or by KeyCode.
   1.3. ***Key:*** Key for respawn.
   1.4. ***Radius:*** Radius the radius in which the player will be respawned.
   1.5. ***Delay:*** Delay before respawn.
   1.6. ***Respawn Health:*** Character health after respawn.
   1.7. ***Respawn Sound:*** Respawn sound will be played after player respawned.
2. **[Screen Fade Properties]**
   2.1. ***In Fade Color:*** Color in fade state.
   2.2. ***In Fade Speed:*** Speed of the transition in fade state.
   2.3. ***Out Fade Speed:*** Speed of the transition out fade state.
   2.4. ***Use Screen Fade:*** If true screen fade on respawn will be active.
   2.5. ***Time To Fade:*** Time to fade complete.

5. Event Trigger

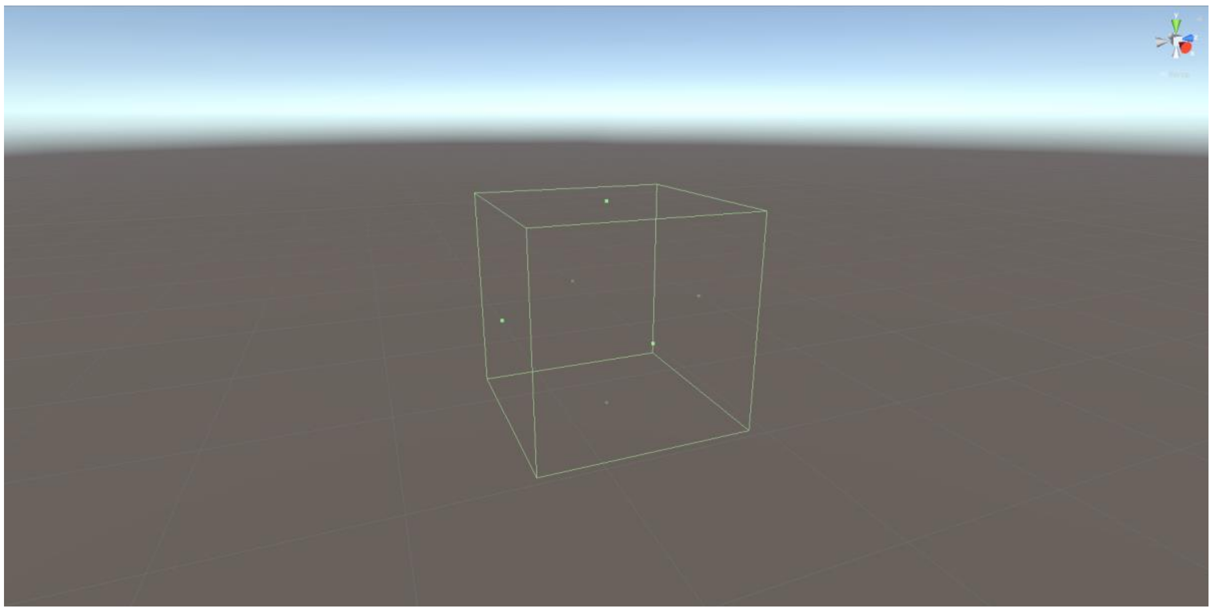
5.1. Create Event Trigger

For create event trigger go to Manager, section “Event Trigger”. Choose trigger shape, you can add custom shape by mesh, set specify a name and press “Create”. After this event trigger will be created.



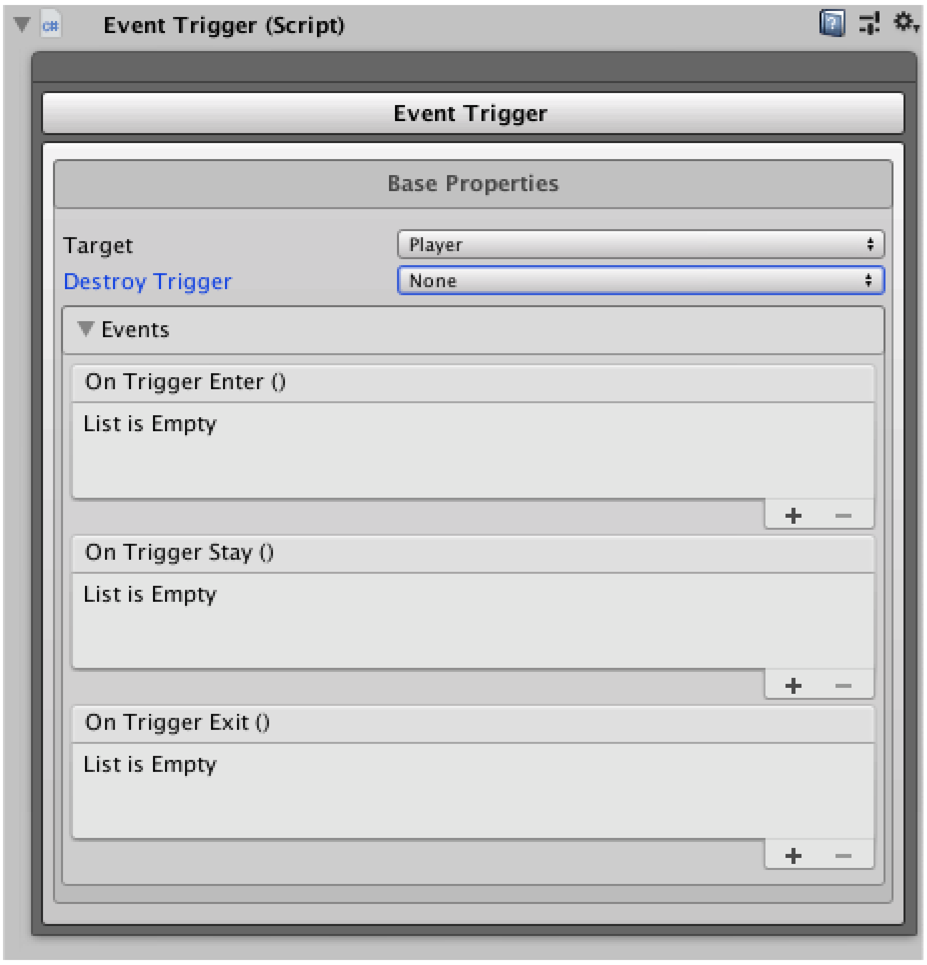
Name: Event Trigger
Trigger Shape: Box

↓



5.2 Event Trigger

All fields of this component have built-in documentation in the editor, just hover the mouse over the required field.


Event Trigger (Script)

Event Trigger

Base Properties

Target: Player

Destroy Trigger: None

Events

On Trigger Enter ()

List is Empty

On Trigger Stay ()

List is Empty

On Trigger Exit ()

List is Empty

Fields Description

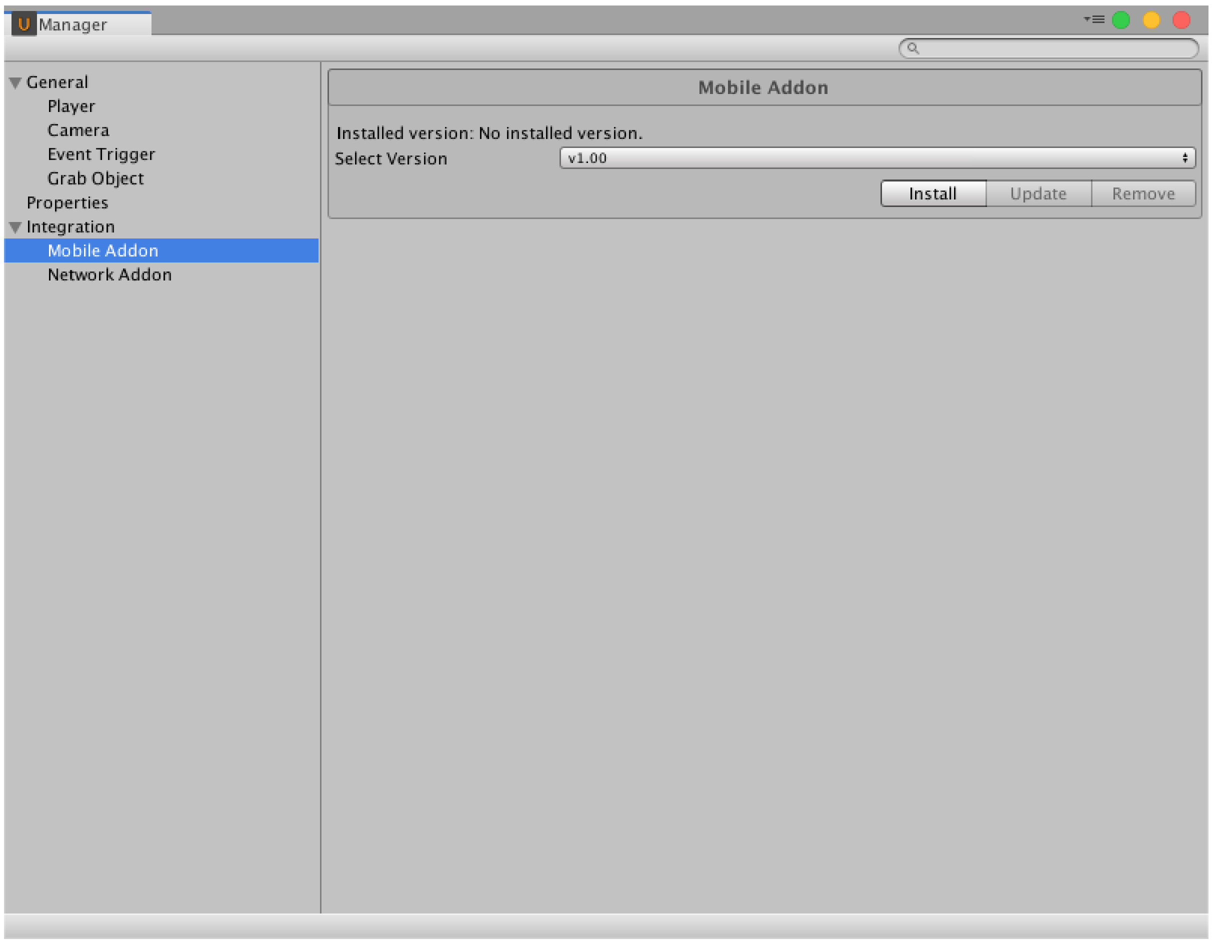
1. **[Base Properties]**
   1.1. ***Target:*** Target layers.
   1.2. ***Destroy Trigger:*** Destroy trigger state.
   1.3. ***Stay Time:*** Destroy trigger after stay specific time in trigger.
   1.4. **[Events]**
      1.4.1. ***OnTriggerEnter:*** Is called when the object with target layer enters in the trigger.
      1.4.2. ***OnTriggerStay:*** Is called when the object with target layer stary in the trigger.

1.4.3. ***OnTriggerExit:*** Is called when the object with target layer exit from the trigger.

6. Mobile Integration

6.1 Install Mobile Addon

For install mobile addon go to Manager, section “Mobile Addon”. Select last version and press “Install” button. Manager automatically install mobile addon in the project. After install is finished, recommend reload Unity Editor.

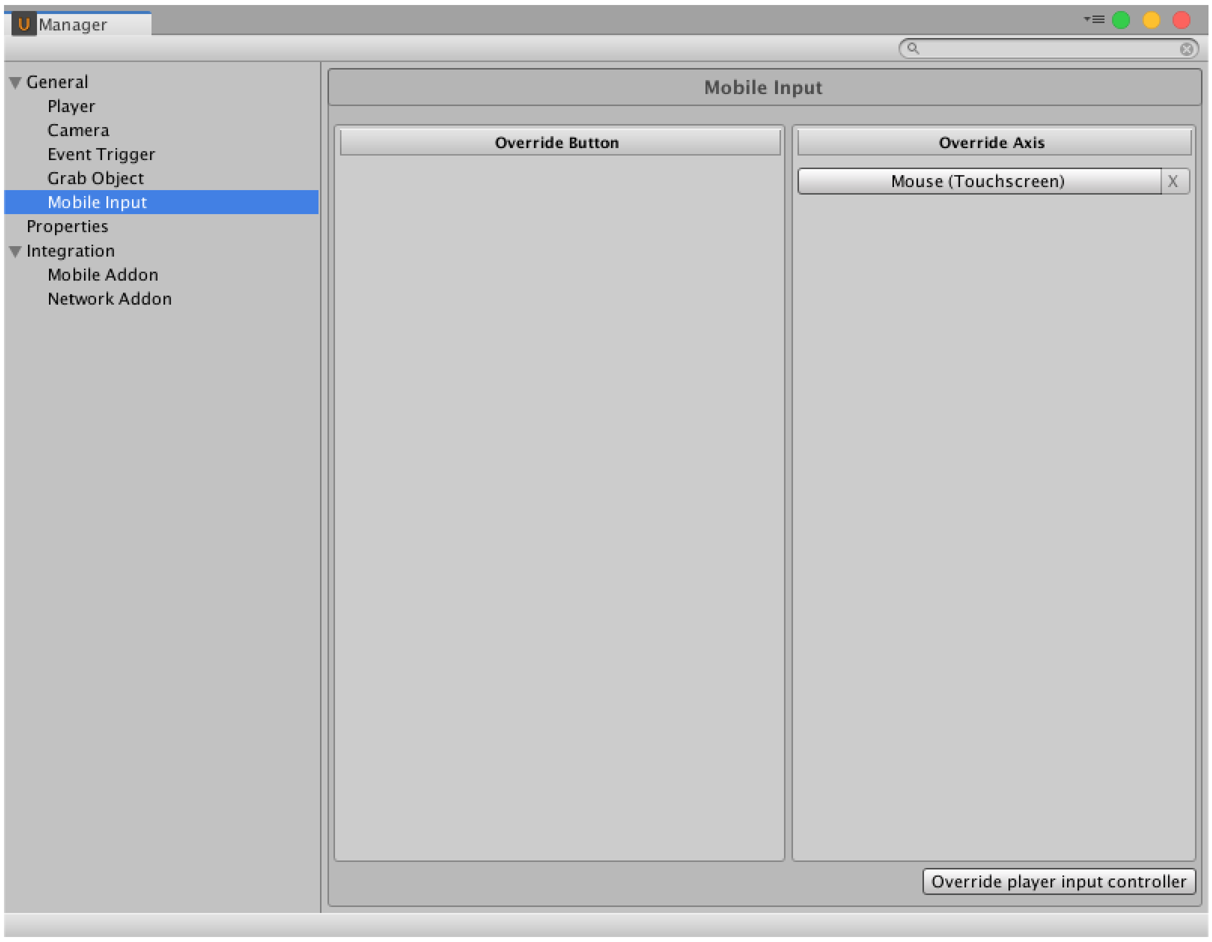


Installed version: No installed version.
Select Version v1.00

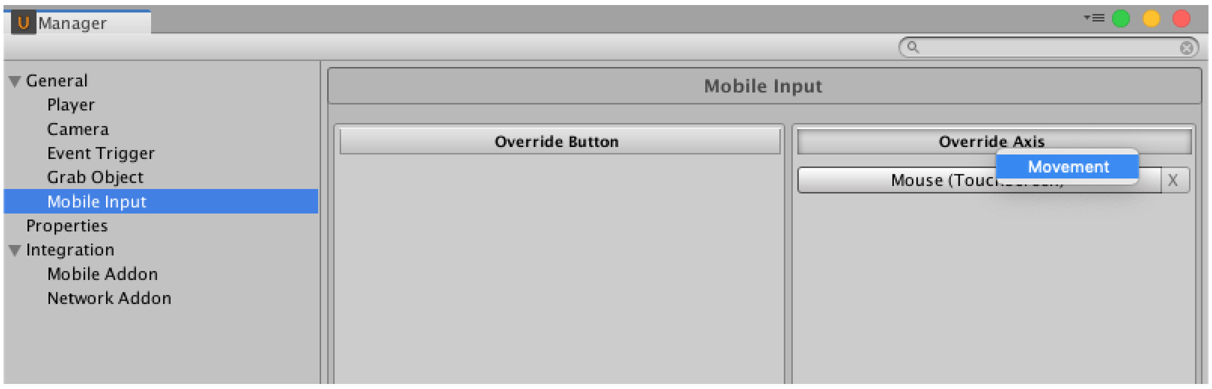
6.2 Override Inputs

For override standalone input go to Manager, section Mobile Input and press “Override player input controller” button. After this Standalone Controller will be overridden on Mobile Controller.

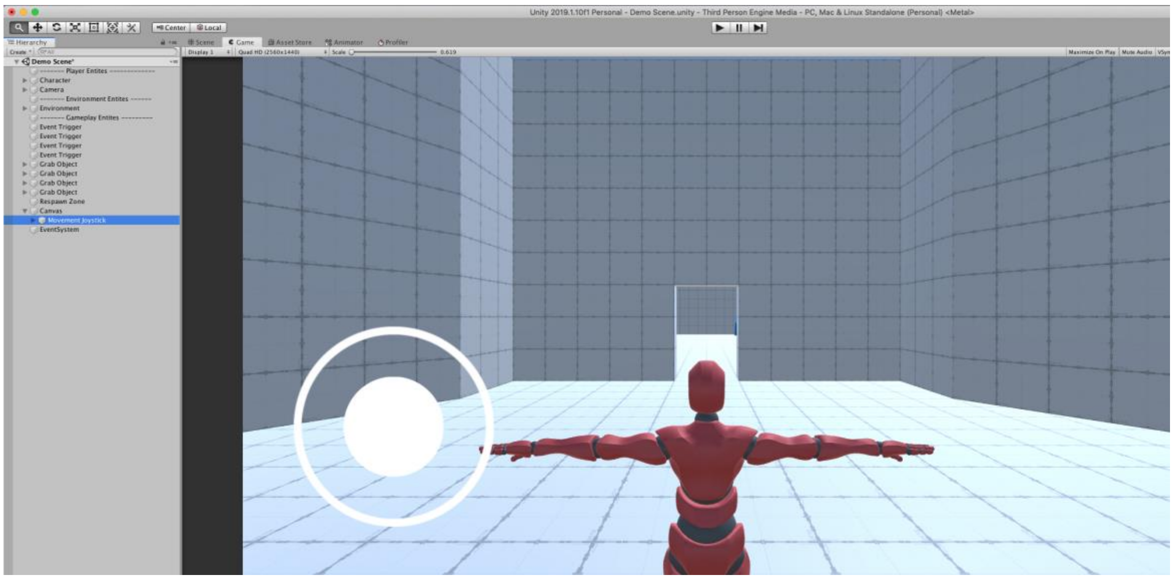
Mouse (Touchscreen) overridden automatically and has a read-only state.



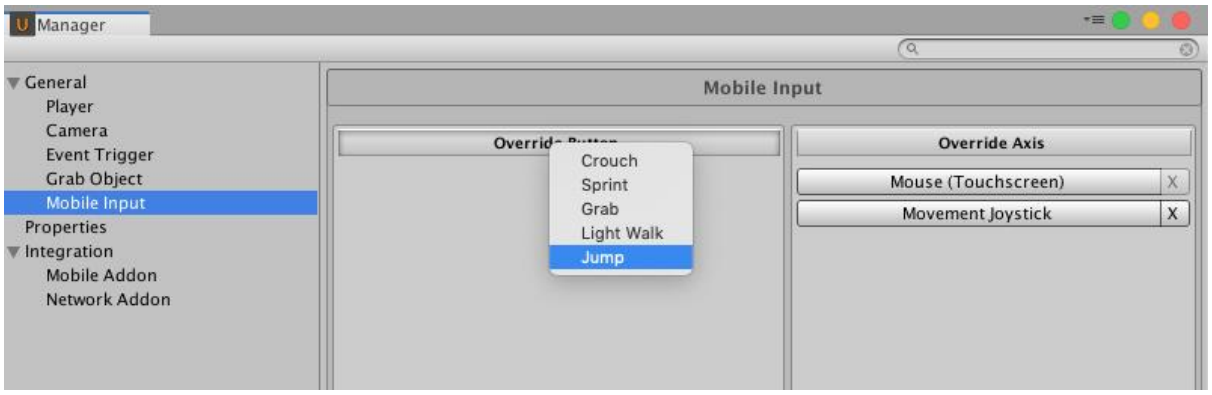
For override movement input press “Override Axis” button and select “Movement”.



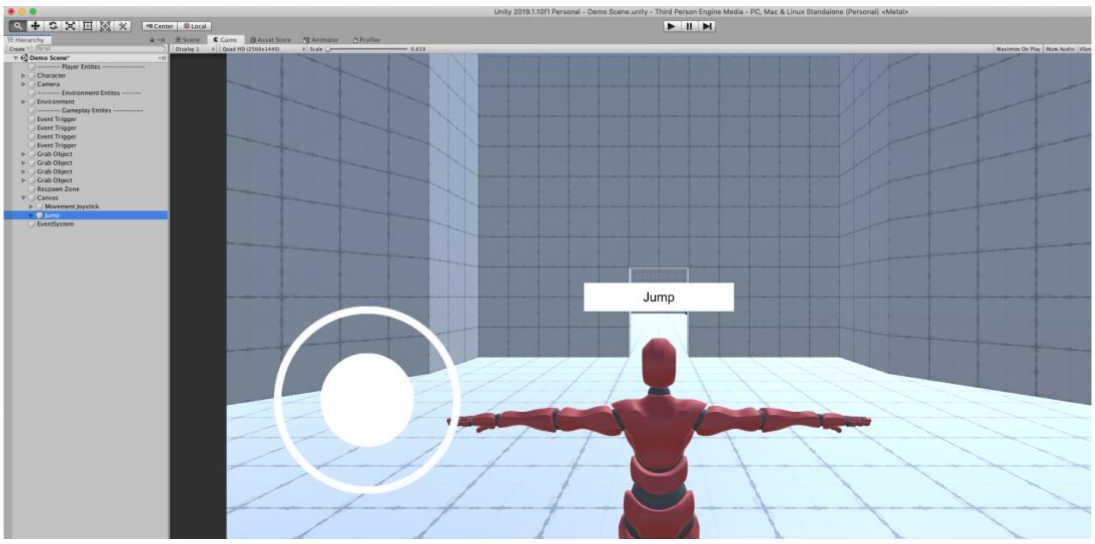
After this on scene create movement joystick. You can completely edit joystick positions, images and settings as you need.


Unity 2019.1.10f1 Personal - Demo Scene.unity - Third Person Engine Media - PC, Mac & Linux Standalone (Personal) <Metal>
Hierarchy
Scene
Game
Asset Store
Animator
Profiler
Demo Scene*
-------- Player Entities --------
Character
Camera
-------- Environment Entities --------
Environment
-------- Gameplay Entities --------
Event Trigger
Event Trigger
Event Trigger
Event Trigger
Grab Object
Grab Object
Grab Object
Grab Object
Respawn Zone
Canvas
Movement Joystick
EventSystem

For override action inputs press “Override Button” button select the action input you want to override.



After this on scene create input button. You can completely edit button positions, images and settings as you need.


Demo Scene*
Character
Camera
Environment
Event Trigger
Grab Object
Respawn Zone
Canvas
Movement Joystick
Jump
EventSystem
Jump