Offline Documentation

💻 Installation

In this page we will cover how to install the package.

Via Package Manager

Open up the **Package Manager** in your Unity project, and search for the FPS Animation Ultimate, Pro, or Lite depending on what tier you own.

Download the package and install it to your project. After that, you should see a **KINEMATION** folder, which contains the FPSAnimationPack asset:

Manually

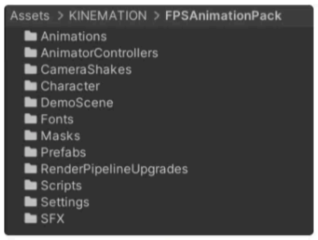
If you already have the unity package, make sure to import it directly to your project. Once the import is done, you should see a **KINEMATION** folder, containing the **FPSAnimationPack** asset.

📚 Overview

In this page we will cover the structure of the package.

Content

All the assets' contents, such as animations, meshes, sounds, and prefabs can be found in **KINEMATION → FPSAnimationPack**:



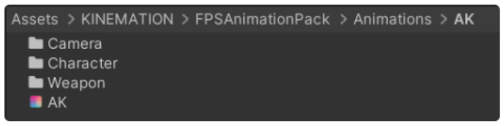
```
Assets > KINEMATION > FPSAnimationPack
  Animations
  AnimatorControllers
  CameraShakes
  Character
  DemoScene
  Fonts
  Masks
  Prefabs
  RenderPipelineUpgrades
  Scripts
  Settings
  SFX
```

Pack content.

Let's break down each directory now.

Animations

This directory contains camera, character, weapon override controllers, and source models. It also includes procedural recoil settings for each weapon.


Assets > KINEMATION > FPSAnimationPack > Animations > **AK**

- Camera
- Character
- Weapon
- AK

AK animations folder example.

Animator Controllers

This directory contains the main animator controllers used to animate characters and weapons. You can find here default animation clips - they are empty and are only used to be overridden by custom animation data.

Camera Shakes

Here you can find settings for different recoil shakes, used to animate the camera when firing.

Character

In this directory, you can find character models, textures, and materials.

Masks

This folder contains Avatar Masks, used in the animation system.

Prefabs

Here you can find prefabs for all weapons, characters, and UI.

Render Pipeline Upgrades

This directory contains unity packages that can upgrade materials to a desired render pipeline. All materials are URP by default.

Scripts

This folder contains gameplay code. All components used in the package are located here.

Settings

Here you can find character and weapon settings, as well as procedural motions (used for aiming and fire mode switching animations).

SFX

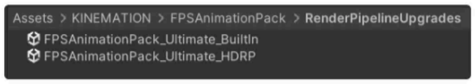
This folder contains all sound effects used in the package.

BiRP, URP and HDRP

In this section we will learn how to upgrade to different render pipelines.

Pack upgrades

The **FPS Animation Pack** supports all three render pipelines in Unity. By default, the imported package is going to be **URP-only**. To upgrade to other render pipelines, go to **KINEMATION → FPSAnimationPack → RenderPipelineUpgrades**:


Assets > KINEMATION > FPSAnimationPack > RenderPipelineUpgrades
FPSAnimationPack_Ultimate_BuiltIn
FPSAnimationPack_Ultimate_HDRP

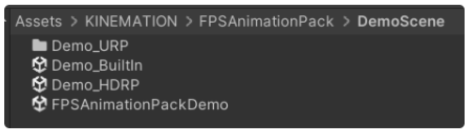
Example upgrade packages.

Select the package for your desired render pipeline and double-click on it. Then, click **"Import"** - this will override weapon and character prefabs with settings adjusted for the selected pipeline.

Note: to revert to *URP*, go to the **Package Manager** and just re-import the asset.

Demo scene upgrades

The packs above only upgrade character arms and weapons. To use a demo scene for your desired render pipeline, make sure to use one of these packages:



Assets > KINEMATION > FPSAnimationPack > DemoScene

- Demo_URP
- Demo_BuiltIn
- Demo_HDRP
- FPSAnimationPackDemo

Demo scene upgrades.

Note: the default scene is **URP**.

Player Prefab

In this section we will cover the character prefab.

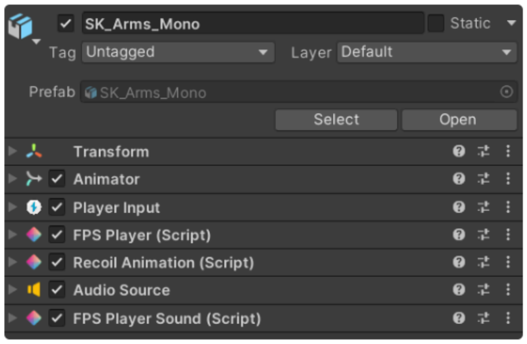
Structure

The **FPSPlayer** prefab is a game-ready entity, that you can drag and drop to the existing level. It consists of 2 parts:

- **SK_Arms_Mono** (Character model).
- **Camera**

Character

Player prefab relies on these components:



SK_Arms_Mono — Static
Tag: Untagged — Layer: Default
Prefab: SK_Arms_Mono

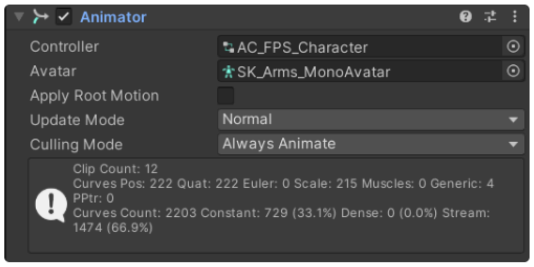
- Transform
- Animator ✓
- Player Input ✓
- FPS Player (Script) ✓
- Recoil Animation (Script) ✓
- Audio Source ✓
- FPS Player Sound (Script) ✓

All character components.

Let's break down each component's settings and purpose.

Animator

It's responsible for applying all animations: reloads, fire, grenade throw, movement, and others.



Animator

Controller	AC_FPS_Character
Avatar	SK_Arms_MonoAvatar
Apply Root Motion	
Update Mode	Normal
Culling Mode	Always Animate

```
Clip Count: 12
Curves Pos: 222 Quat: 222 Euler: 0 Scale: 215 Muscles: 0 Generic: 4
PPtr: 0
Curves Count: 2203 Constant: 729 (33.1%) Dense: 0 (0.0%) Stream:
1474 (66.9%)
```

Animator component.

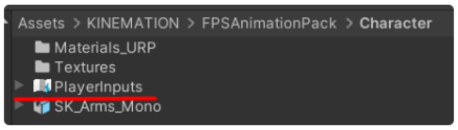
AC_FPS_Character is the main character controller. It is used as a source for all override controllers in the package.

Tip: this controller contains empty animation clips, overridden by weapon-specific animations.

Because it is a part of the animation system, it will be covered later in the **Character** section.

Player Input

This is a component from Unity's *New Input System* package. You can find the input map here:



Assets > KINEMATION > FPSAnimationPack > **Character**

- Materials_URP
- Textures
- PlayerInputs
- SK_Arms_Mono

Player inputs.

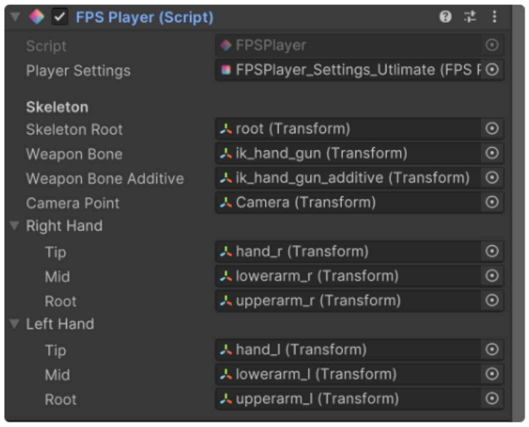
Note: the **FPS Animation Pack** supports legacy inputs as well, so it will work even if the *New Input System* is not imported.

The package comes with these gameplay actions:

1. **Reload** (R).
2. **Fire** (LMB).
3. **Aim** (RMB).
4. **Change Weapon** (F).
5. **Throw Grenade** (G).
6. **Move** (WASD).
7. **Sprint** (Left Shift).
8. **Tactical Sprint** (Left Shift + X).
9. **Change Fire** Mode (B).
10. **Mouse Wheel** (instant weapon swapping).

FPS Player

This component is responsible for weapon management and procedural animation:



FPS Player (Script)

Script	FPSPlayer
Player Settings	FPSPlayer_Settings_Utlimate (FPS F
Skeleton	
Skeleton Root	root (Transform)
Weapon Bone	ik_hand_gun (Transform)
Weapon Bone Additive	ik_hand_gun_additive (Transform)
Camera Point	Camera (Transform)
Right Hand	
Tip	hand_r (Transform)
Mid	lowerarm_r (Transform)
Root	upperarm_r (Transform)
Left Hand	
Tip	hand_l (Transform)
Mid	lowerarm_l (Transform)
Root	upperarm_l (Transform)

FPS Player component.

- **Player Settings**: contains character-related properties.
- **Skeleton Root**: the model root bone.
- **Weapon Bone**: the weapon bone, all weapons are parented to it.
- **Weapon Bone Additive**: the bone used to apply additive movement.
- **Camera Point**: it will be used to align iron sights.
- **Right/Left Hand**: used for applying procedural animations.
    - Tip: the hand bone.
    - Mid: the elbow bone.
    - Root: the upper arm bone.

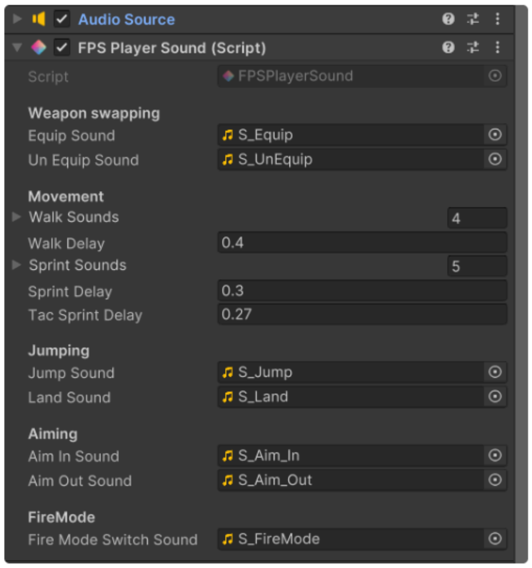
Note: the procedural animation is running in *LateUpdate()*.

Recoil Animation

This component generates procedural firing animations. You can find out more here.

Audio Source and FPS Player Sound

These components are used together to play general character effects, such as walking, swapping weapons, or throwing a grenade:



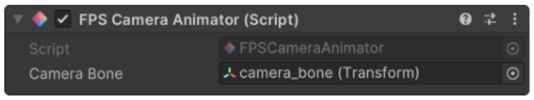
Audio Source
FPS Player Sound (Script)

Script	FPSPlayerSound
Weapon swapping	
Equip Sound	S_Equip
Un Equip Sound	S_UnEquip
Movement	
Walk Sounds	4
Walk Delay	0.4
Sprint Sounds	5
Sprint Delay	0.3
Tac Sprint Delay	0.27
Jumping	
Jump Sound	S_Jump
Land Sound	S_Land
Aiming	
Aim In Sound	S_Aim_In
Aim Out Sound	S_Aim_Out
FireMode	
Fire Mode Switch Sound	S_FireMode

FPS Player Sounds.

Camera

Custom camera animations and recoil shakes are applied via the **FPS Camera Animator** component:



FPS Camera Animator (Script)

Script	FPSCameraAnimator
Camera Bone	camera_bone (Transform)

Camera animator.

Note: camera curves exist in character animations, so whenever a player clip is played - it will also animate the camera.

Weapon System

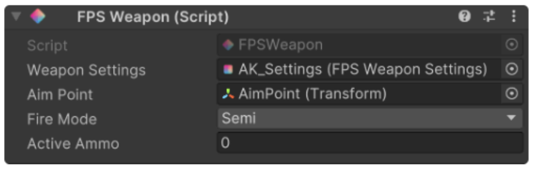
In this section we will cover the weapon system.

Overview

Weapons are responsible for these gameplay actions:

1. Reloading
2. Changing fire modes
3. Equip/UnEquip animations
4. Firing

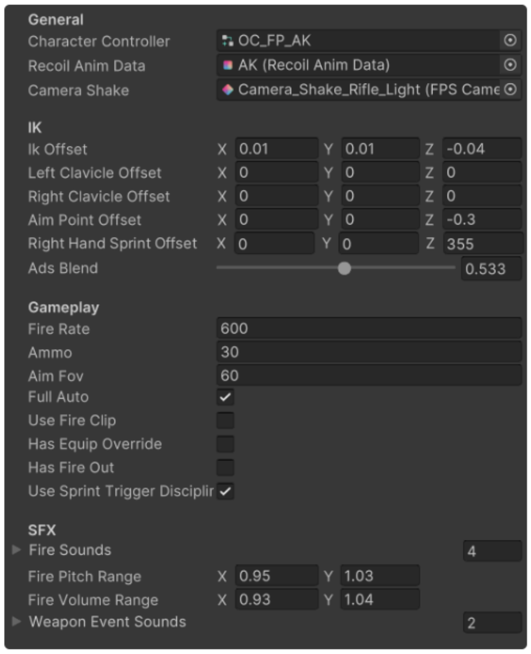
The functions above are implemented in the **FPSWeapon** component:



FPS Weapon (Script)

Script	FPSWeapon
Weapon Settings	AK_Settings (FPS Weapon Settings)
Aim Point	AimPoint (Transform)
Fire Mode	Semi
Active Ammo	0

FPSWeapon component.



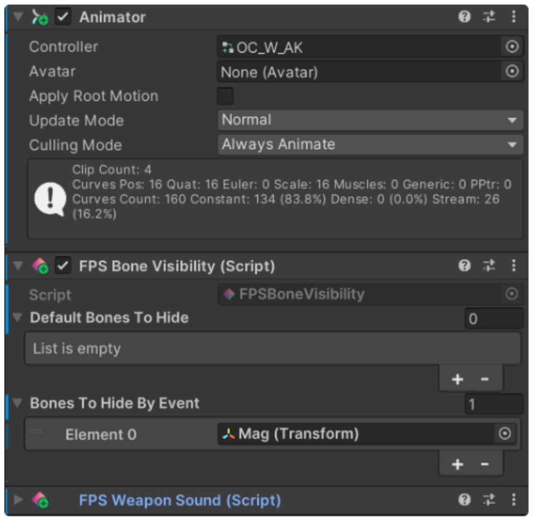
General			
Character Controller	OC_FP_AK		
Recoil Anim Data	AK (Recoil Anim Data)		
Camera Shake	Camera_Shake_Rifle_Light (FPS Came		
IK			
Ik Offset	X 0.01	Y 0.01	Z -0.04
Left Clavicle Offset	X 0	Y 0	Z 0
Right Clavicle Offset	X 0	Y 0	Z 0
Aim Point Offset	X 0	Y 0	Z -0.3
Right Hand Sprint Offset	X 0	Y 0	Z 355
Ads Blend	0.533		
Gameplay			
Fire Rate	600		
Ammo	30		
Aim Fov	60		
Full Auto	✓		
Use Fire Clip			
Has Equip Override			
Has Fire Out			
Use Sprint Trigger Disciplir	✓		
SFX			
Fire Sounds	4		
Fire Pitch Range	X 0.95	Y 1.03	
Fire Volume Range	X 0.93	Y 1.04	
Weapon Event Sounds	2		

Weapon settings.

1. **Character Controller**: override controller for this weapon.
2. **Recoil Anim Data**: procedural recoil settings.
3. **Camera Shake**: recoil camera shakes.
4. **Ik Offset**: translation offset for the arms.
5. **Left Clavicle Offset**: translation offset for the left arm.
6. **Right Clavicle Offset**: translation offset for the right arm.
7. **Aim Point Offset**: translation offset for the aiming socket.
8. **Right Hand Sprint Offset**: rotation offset activated when tactical sprinting.
9. **Ads Blend**: blend between absolute and additive aiming.
10. **Fire Rate**: in rounds-per-minute.
11. **Ammo**: default ammo size.
12. **Aim Fov**: field-of-view when aiming.
13. **Full Auto**: whether the weapon supports full auto.
14. **Use Fire Clip**: whether to use firing animation.
15. **Has Equip Override**: whether to use custom equip animation.
16. **Has Fire Out**: whether to lock the bolt after firing.
17. **Use Sprint Trigger Discipline**: whether to use gun safety when sprinting.
18. **Fire Sounds**: firing sounds played randomly.
19. **Fire Pitch Range**: range for the pitch.
20. **Fire Volume Range**: range for the volume.
21. **Weapon Event Sounds**: reloading sounds played via animation events.

Weapon mesh

The weapon model has these components:



Model components.

Animator is used to animate gun parts (bolt, mag, etc.), which are automatically synced with the character's movements.

FPS Bone Visibility is responsible for hiding or unhiding bones, based on animation events.

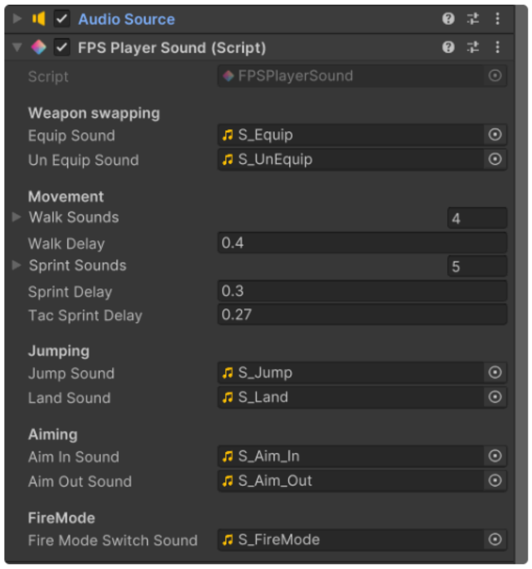
FPS Weapon Sound plays sounds related to the weapon (e.g. reloading, firing).

Sound Effects

Page description (optional)

Overview

All sound effects are played via the **Audio Source** component attached to the character:



Audio Source

FPS Player Sound (Script)

Script	FPSPlayerSound
Weapon swapping	
Equip Sound	S_Equip
Un Equip Sound	S_UnEquip
Movement	
Walk Sounds	4
Walk Delay	0.4
Sprint Sounds	5
Sprint Delay	0.3
Tac Sprint Delay	0.27
Jumping	
Jump Sound	S_Jump
Land Sound	S_Land
Aiming	
Aim In Sound	S_Aim_In
Aim Out Sound	S_Aim_Out
FireMode	
Fire Mode Switch Sound	S_FireMode

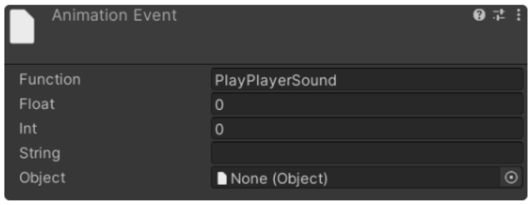
Audio Source.

FPS Player Sound component is used to play general sounds, not specific to any weapon. Other sounds, like reloads or firing, are played by the weapons - they access the character's Audio Source and play the target sound effect.

Animation Events

Many sounds, like reloads, aiming, walking, or fire mode switching are played via animation events. Events are added directly to the *Animation Clip* and then are automatically triggered in runtime. There are 2 types of animation events.

PlayPlayerSound

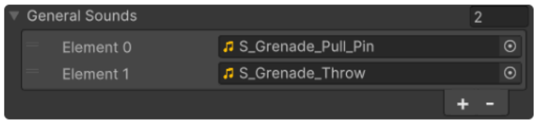


Animation Event

Function	PlayPlayerSound
Float	0
Int	0
String	
Object	None (Object)

Player sound event.

This event has an input index parameter, which defines the index of the target sound. In this case, sounds will be selected from this list:



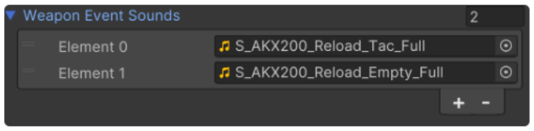
General Sounds 2

Element 0	S_Grenade_Pull_Pin
Element 1	S_Grenade_Throw

FPS Player Settings.

PlayWeaponSound

This event is added to reloading animations. Sounds are selected from:



Weapon Event Sounds 2

Element 0	S_AKX200_Reload_Tac_Full
Element 1	S_AKX200_Reload_Empty_Full

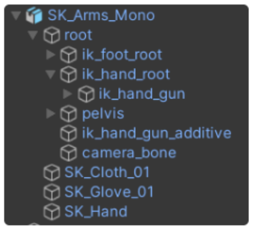
FPS Weapon Settings.

Character

In this chapter we will cover character animation aspect.

Skeleton Structure

The FPS arms model used in the package is based on UE4 Skeleton:



```
SK_Arms_Mono
  root
    ik_foot_root
    ik_hand_root
      ik_hand_gun
    pelvis
    ik_hand_gun_additive
    camera_bone
  SK_Cloth_01
  SK_Glove_01
  SK_Hand
```

Character skeleton.

This rig includes two important bones:

1. **ik_hand_gun** - contains the baked movement of the weapon.
2. **ik_hand_gun_additive** - used for procedural idle, walking, and sprinting animations.
3. **camera_bone** - contains camera keyframes.

Tip: the character rig type is Generic.

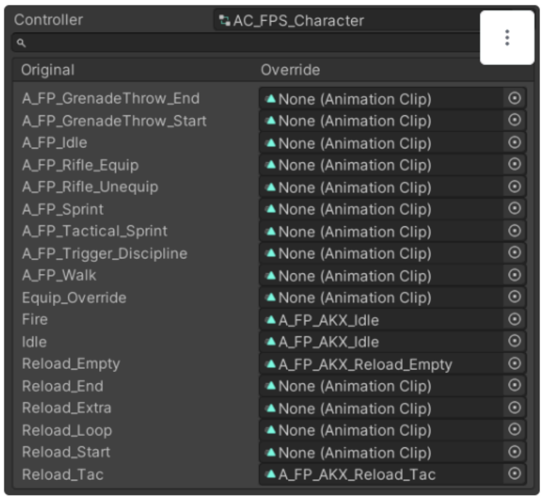
All animations also include keyframes for the lower body, meaning that clips can be used with full-body characters.

Override Animator Controllers

All character animations are defined in override controllers. The **FPS Animation Pack** uses a convenient naming convention:

> **OC_FP_WeaponName, OC stands for "Override Controller"**

All character controllers have the same structure:



Controller: AC_FPS_Character

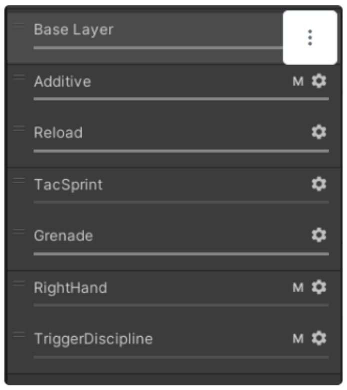
Original	Override
A_FP_GrenadeThrow_End	None (Animation Clip)
A_FP_GrenadeThrow_Start	None (Animation Clip)
A_FP_Idle	None (Animation Clip)
A_FP_Rifle_Equip	None (Animation Clip)
A_FP_Rifle_Unequip	None (Animation Clip)
A_FP_Sprint	None (Animation Clip)
A_FP_Tactical_Sprint	None (Animation Clip)
A_FP_Trigger_Discipline	None (Animation Clip)
A_FP_Walk	None (Animation Clip)
Equip_Override	None (Animation Clip)
Fire	A_FP_AKX_Idle
Idle	A_FP_AKX_Idle
Reload_Empty	A_FP_AKX_Reload_Empty
Reload_End	None (Animation Clip)
Reload_Extra	None (Animation Clip)
Reload_Loop	None (Animation Clip)
Reload_Start	None (Animation Clip)
Reload_Tac	A_FP_AKX_Reload_Tac

AKX200 override controller.

"A_FP"-type animations are not typically overridden, as these are general motions used for all guns. **Fire**, **Idle**, and **Reload** slots are overridden depending on the weapon type. Now it's time to cover the main **AC_FPS_Character** controller.

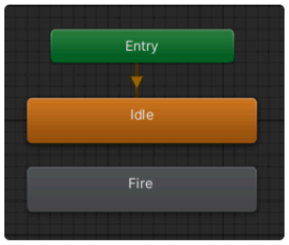
Main Character Controller

AC_FPS_Character consists of 7 layers:



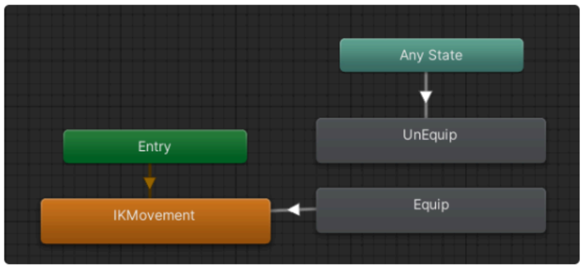
Animation layers.

Base Layer



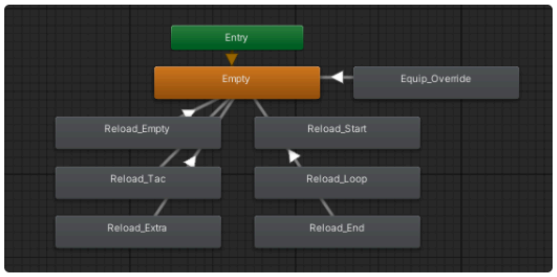
This layer includes the base idle pose and firing state.

Additive Layer



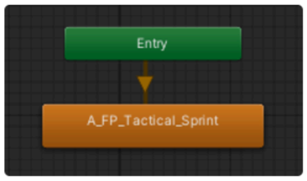
This layer only animates the **ik_hand_gun_additive** bone. It is used later to apply IK animation on top of the character base pose. This allows to re-use of one set of animations for all weapons.

Reload Layer



This layer is used to play reloading animations. It also has a slot for **Equip_Override** - it is used exclusively for the Kolibri equip animation.

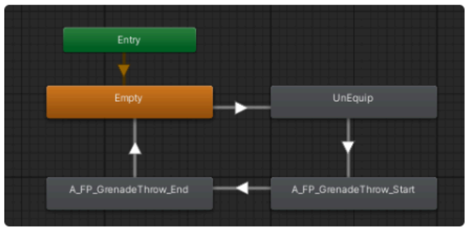
Tac Sprint Layer



This layer only applies tactical sprinting animation.

Grenade Layer

This layer is used to play the grenade throw sequence:



First, the character unequips the current weapon, then throws the grenade and gets back to the idle state.

Right Hand and Trigger Discipline

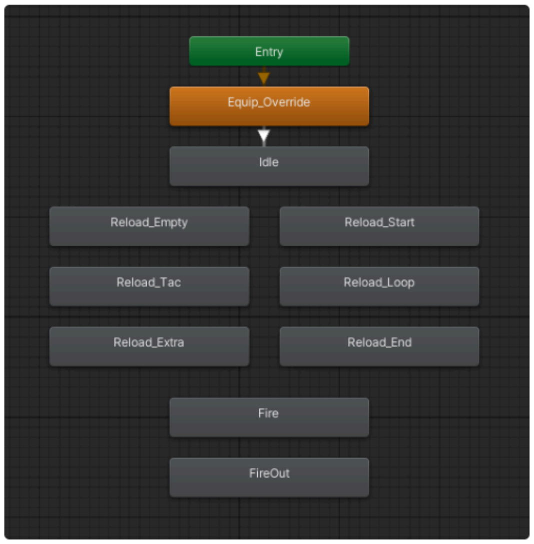
These layers are activated when a character is tactical sprinting.

🔫 Weapon

In this chapter we will cover animation aspect of the weapon.

Animator Controllers

All weapon animator controllers have the same structure:



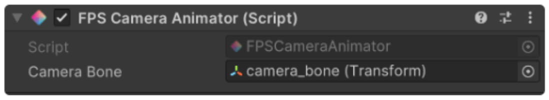
AC_FPS_Weapon

Each override controller is assigned to the weapon animator directly.

📷 Camera

In this page we will cover the camera animation aspect.

The camera animations are already baked into the character animation keyframes for the **camera_bone**. Additionally, the package includes a recoil shake system implemented in the **FPS Camera Animator**:



FPS Camera Animator (Script)

Script	FPSCameraAnimator
Camera Bone	camera_bone (Transform)

Camera Animator.