Unity Animation System

A static world seems lifeless, and to create an immersive 3D experience, using animation becomes vital. Take a look at these two gifs and decide which one looks alive.

Trust me, this is a moving gif.

You can see movement here.

When you look at the animations of a video game, you can see the effort developers put into making the character walk, run, jump, and even during inactive states! The images above show the difference between the GameObject being static and it being alive.

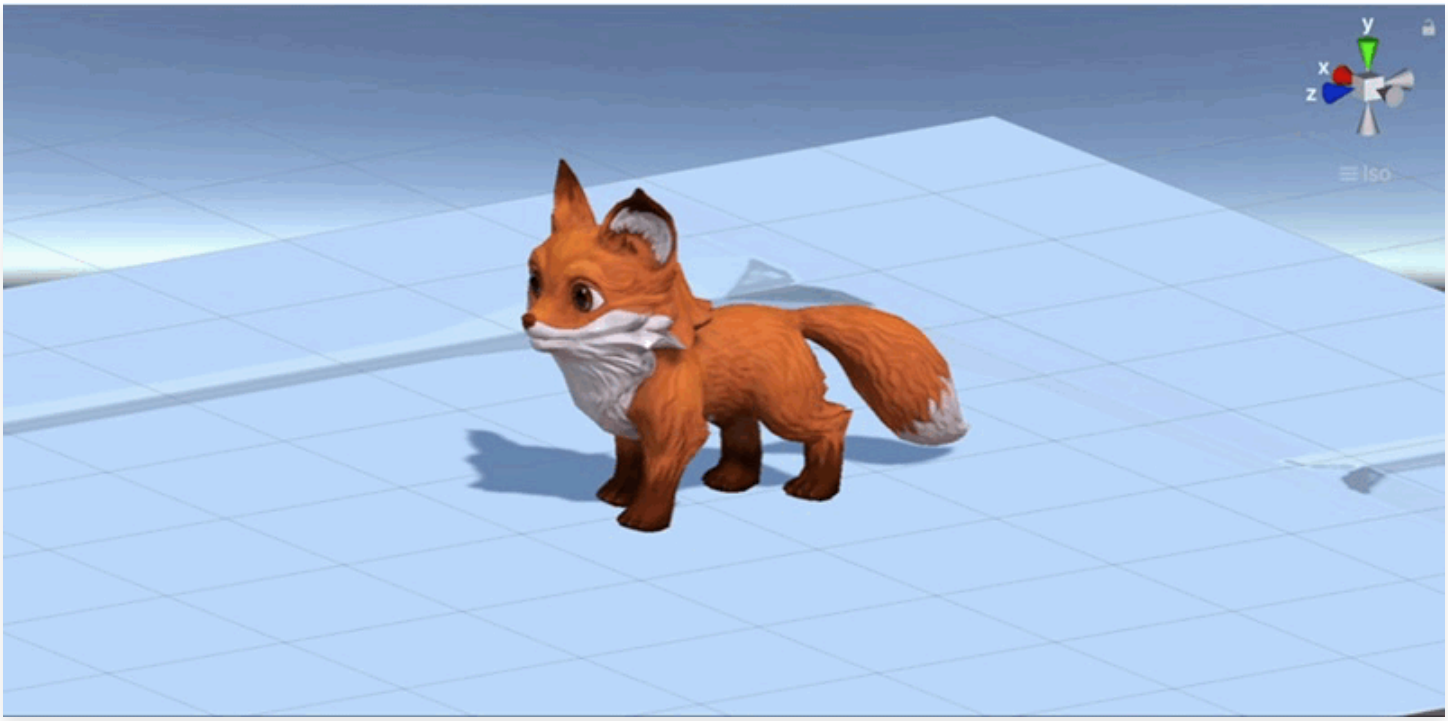
To create and manage animations, depending on what a GameObject should look like, Unity has its **Animation System** that provides simple workflows to animate your 3D experiences. You may hear its name as Mecanim; it is the same system. Different engines work in different ways. In Unreal, the animation system is called Blendspace. It has a similar inner working, so understanding these concepts is a good start.

In VR, character animations, the player's hands, and how things move can create or break immersion. For this reason, it is necessary to understand the Animation System.

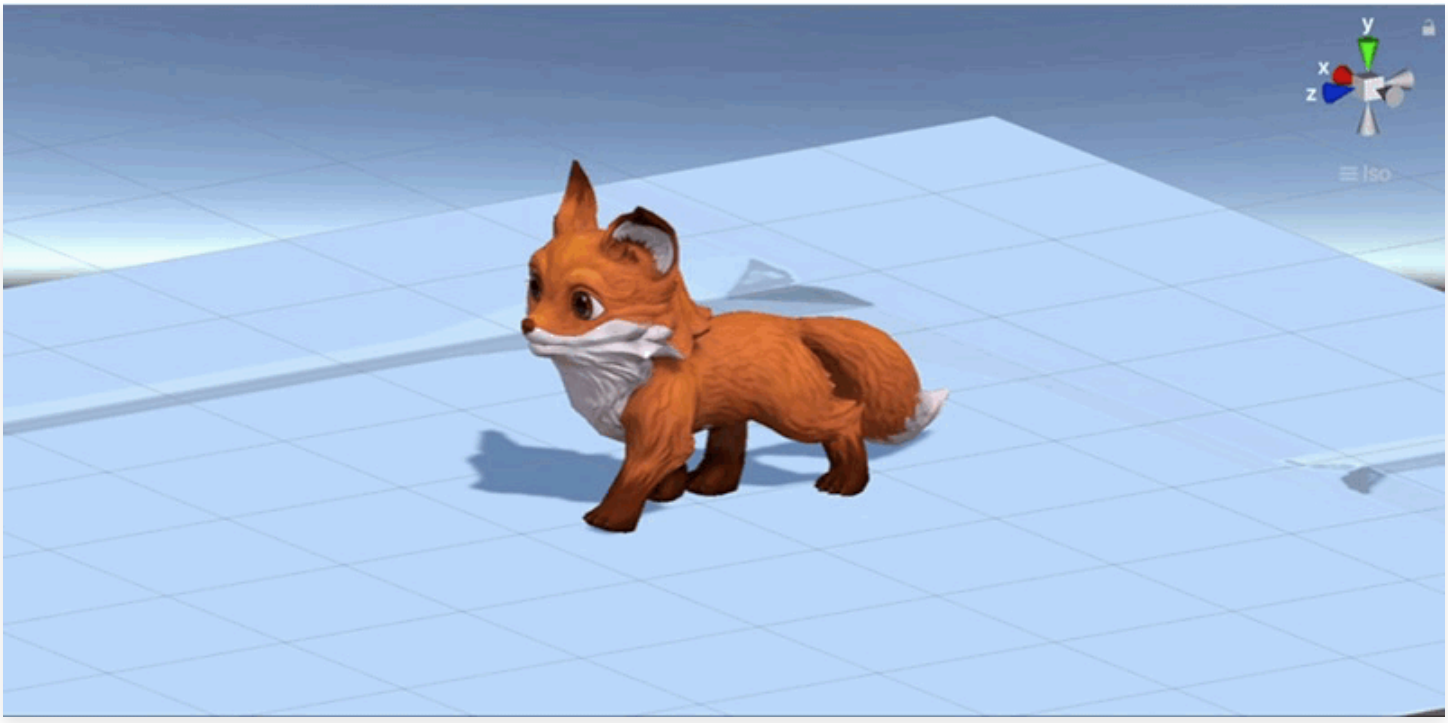
Read the manual: Manual: Animation System Overview | Unity↗

Animation Clips

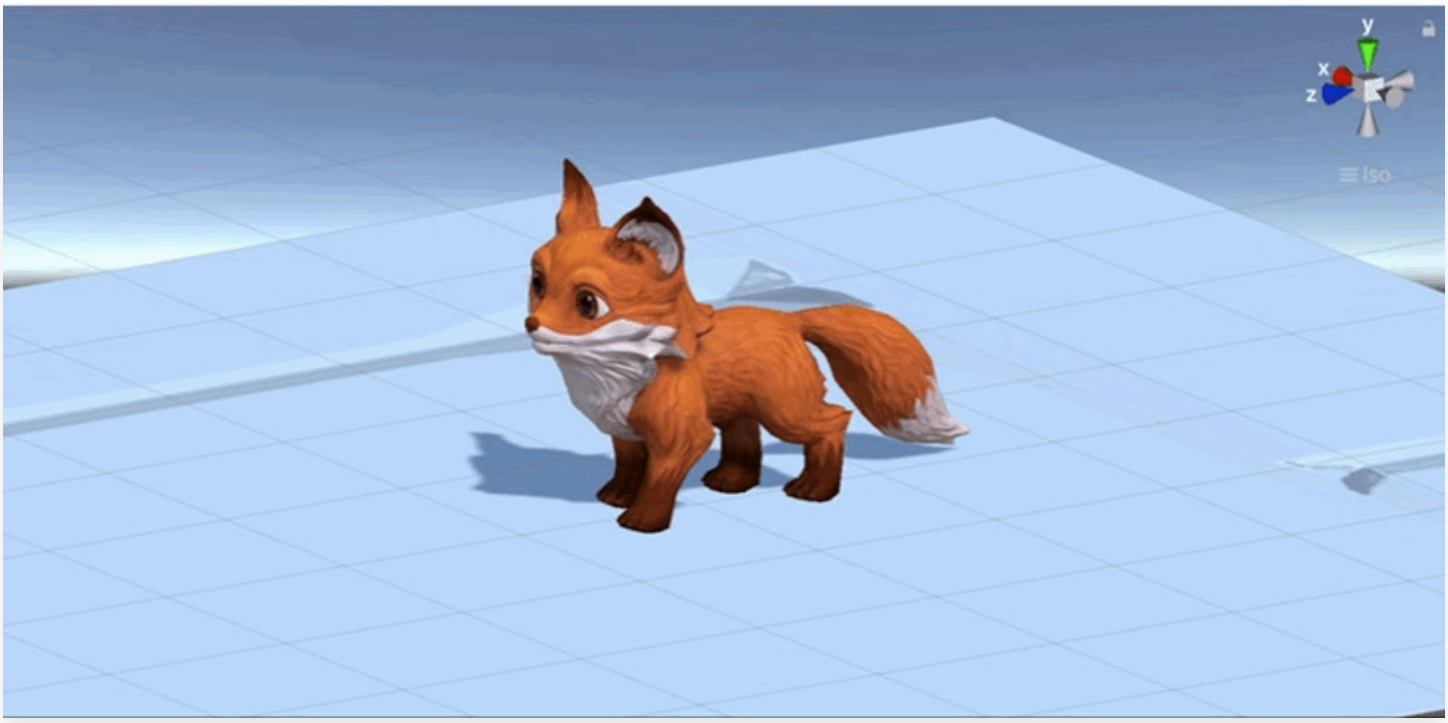
Animation clips are pieces of motion that can be strung together. For example, the *Fox_Idle* animation clip makes it move a little bit. The fox remains still, but the animation clip adds life to it.



You can (and should) have more than one clip: *Fox_Walk_InPlace* adds even more life and makes the fox walk!



Fox_Idle and *Fox_Walk_InPlace* are looping animation clips and can indefinitely run. A third one that is quite common is a jump animation. See the *Fox_Jump* animation clip below.



Animation clips contain information about position and movements that Unity uses to create a compelling experience. You can import animation clips from external sources like the Unity Asset Store↗, or create your own using the Animation window.

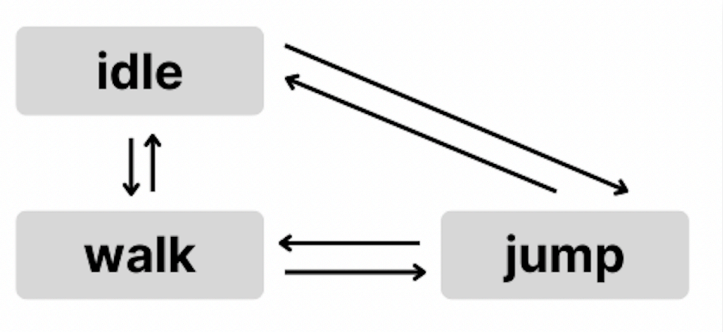
Read more about it: Manual: Animation Clips | Unity↗

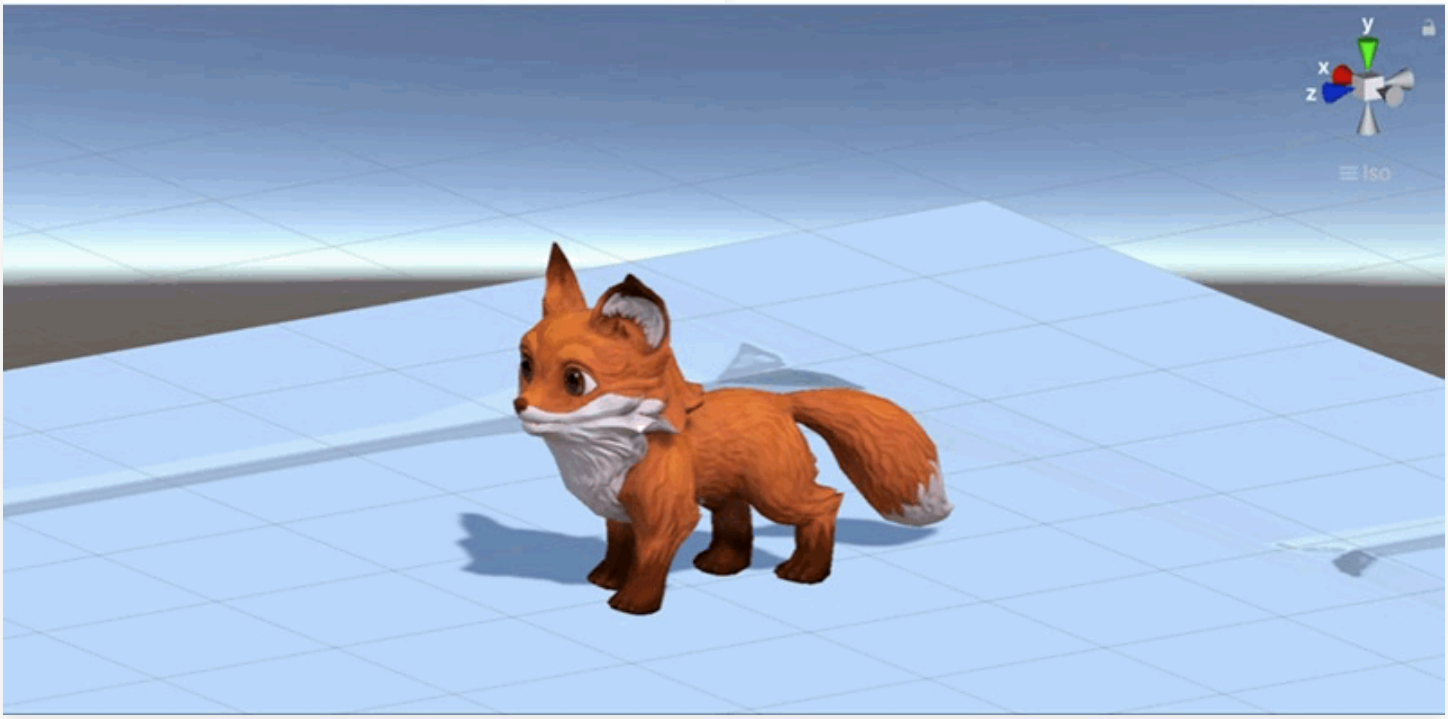
Animation States

Purposefully, the name of the animation clips: *Fox_Idle*, *Fox_Walk_InPlace*, and *Fox_Jump* explain three states of the fox. It can have an idle state, walk, or jump. Just like naming scripts, variables, and functions, being descriptive with names is a big help.

You could create an experience where the fox changes between these states, and therefore, the fox should transition between animation clips and go from any of these to another one. A good graphic could be this one:

In this case, the default state is idle and, if nothing is happening, the animation clip Fox_Idle should play on a loop. When an action happens, the fox moves to the appropriate animation state: walk or jump. Which comes with the corresponding animation clip: *Fox_Walk_InPlace* or *Fox_Jump*.





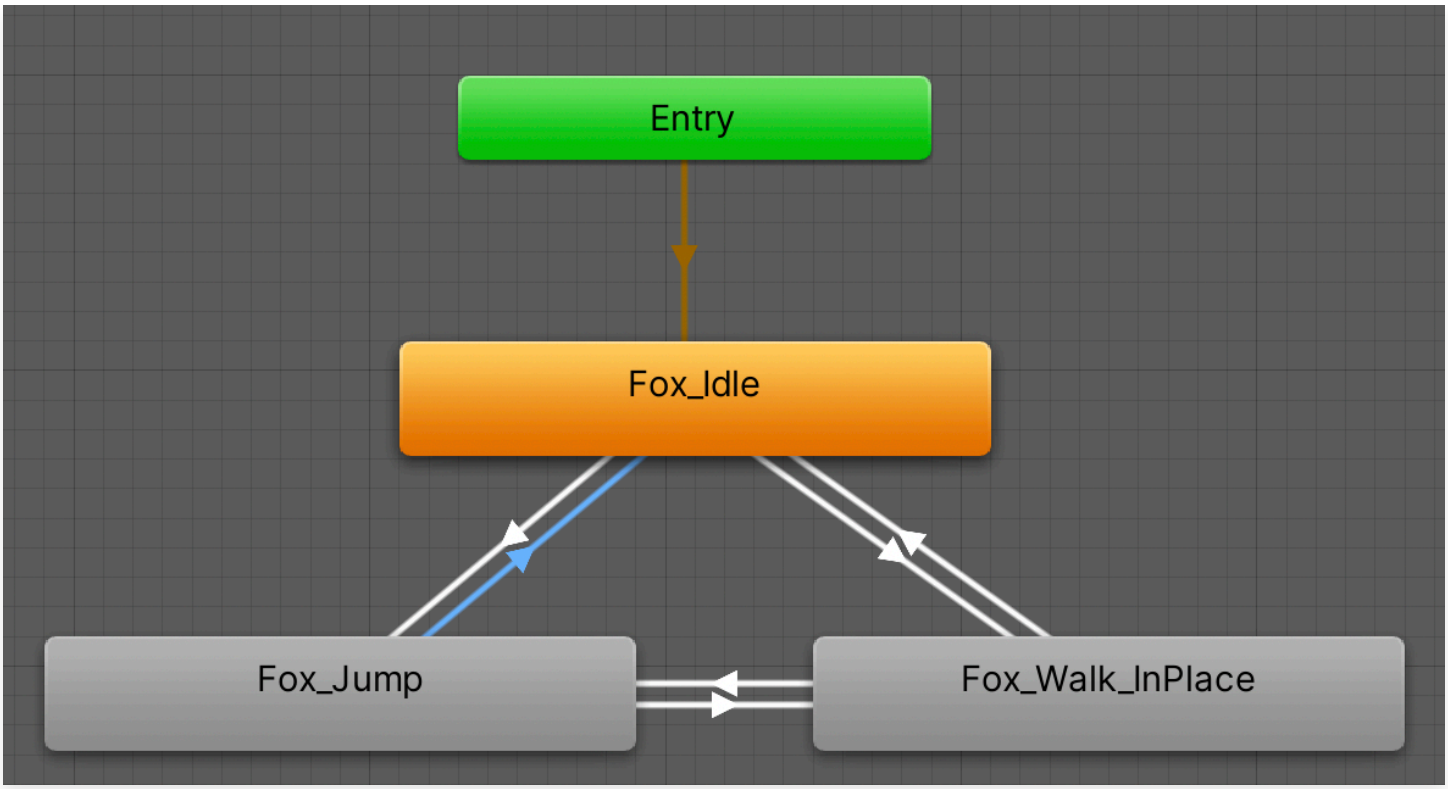
The Animation Controller, part of the Animator component, is the tool to manage animation states and clips.

State Machines

Unity Animation System displays a flow-chart type of graphic representing a **state machine**.

Read about it: Manual: State Machine Basics | Unity↗

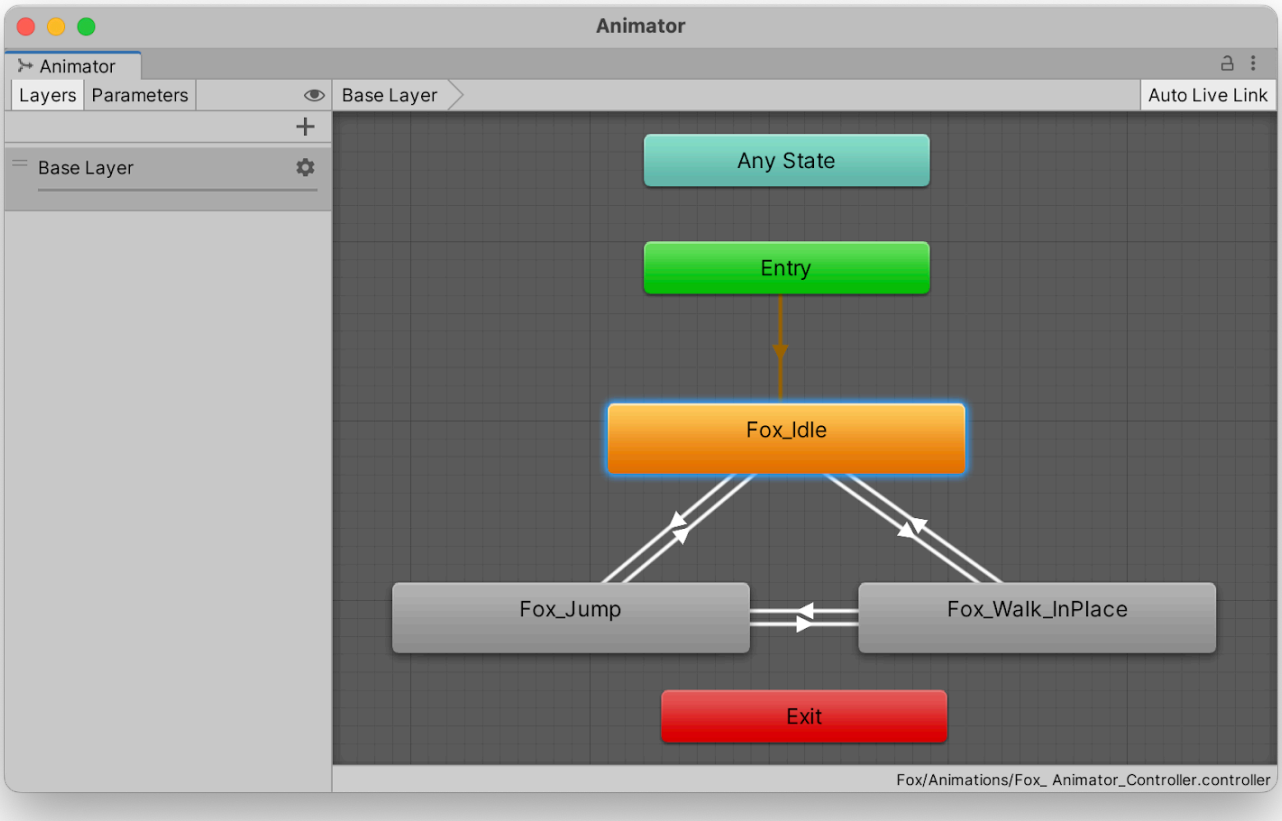
A State Machine shows the different possible states of a GameObject, and how it can transition from one to the other. In the graph, the states are nodes, and the arrows between them are the transitions that determine how the GameObject can go from one state to the other. The Animator window allows the developers to create sequences without coding.



The Animator Controller

An Animator Controller is a file that contains references to the animation states and clips to manage them in what can be considered a flowchart. It is all programming, but you can see something like this:

The Animator Controller represents the states in the form of boxes and transitions represented by arrows between the states. In this window, you can customize the behavior of GameObjects, and reuse them for different GameObjects. You can change animation states with triggers in UnityEvents or code.



The animation controller lives in the Animator component, which you will learn to use in this course.

Read the documentation: Manual: Animator Controller | Unity ↗
Manual: Animator Component | Unity ↗

Summary

A GameObject can contain an Animator component where you can attach an Animator Controller. The controller allows you to manage the different animation states and animation clips.

The different animation clips play and transition depending on the animation states of the GameObject.
