AnimationClipPlayable

窗体顶端

窗体底端

窗体顶端

1. [**pgeorges-dpt**](https://forum.unity.com/members/pgeorges-dpt.1095501/)

[](https://forum.unity.com/members/pgeorges-dpt.1095501/)

Joined:

Apr 7, 2016

Posts:

[43](https://forum.unity.com/search/member?user_id=1095501)

Hey there,  
  
What am I supposed to do with AnimationClipPlayable now that it has disappeared in 5.6?  
  
Cheers,  
  
Paul

[pgeorges-dpt](https://forum.unity.com/members/pgeorges-dpt.1095501/), [**Jan 18, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/)

[#1](https://forum.unity.com/threads/animationclipplayable.451854/)

1. [**pierrepaul**](https://forum.unity.com/members/pierrepaul.112457/)

[](https://forum.unity.com/members/pierrepaul.112457/)

Unity Technologies

Joined:

Jun 19, 2012

Posts:

[161](https://forum.unity.com/search/member?user_id=112457)

Hi.  
  
It has not disappeared, the Playable API has changed in 5.6 and the docs will be updated real soon.  
  
until then, here is the draft of the documentation  
  
  
Playable API  
  
  
The Playables API provides a way to create tools, effects or other gameplay mechanisms by organizing and evaluating data sources in a tree-like structure.  
  
  
This tree-like structure allows you to mix/blend/modify multiple data sources and play them through a single output. In its current form, the Playable API supports animation graphs, providing the capacity to interact with Mecanim via scripting.  
  
  
Even though the playable api is currently limited to AnimationPlayable and ScriptPlayable, it is a generic API that will in the future be applied to Audio, Video and other systems.  
  
  
**How does it work**  
  
**The PlayableGraph, Playable, PlayableHandle and PlayableOutput**  
  
The PlayableGraph is responsible for lifecycle of the Playables. It it used for creation, destruction and connection of Playables. The PlayableGraph defines a set of PlayableOutputs that are binded to scene object/component. For instance, an AnimationOutput is binded to to an Animator Component.  
  
  
Generally, when working with the graph, it is preferred to use PlayableHandles. Theses C# struct have the advantage of not allocating GC memory. PlayableHandles are used for all topological operations - connection, disconnection, creation - on the graph. It is always possible to get the Playable object from the PlayableHandle by using the PlayableHandle.GetObject<> method. The Playable.handle method returns the PlayableHandle of the Playable.  
  
  
Playable Lifetime  
  
  
Destroying the PlayableGraph will automatically destroy all Playables and PlayableOutputs that were created by the graph. Users must call PlayableGraph.Destroy() manually otherwise Unity will generate an error message.  
  
  
**Examples**  
**Playing a single clip on a GameObject**  
  
The following MonoBehaviour creates a simple tree with a single node, which plays a single clip.  
  
  
A new type of object is introduced in this example: the AnimationClipPlayable, which derives from the generic AnimationPlayable, and wraps an AnimationClip in order to make it compatible with the Playable API.  
  
  
using UnityEngine;  
using UnityEngine.Experimental.Director;  
  
[RequireComponent (typeof (Animator))]  
public class PlayAnimation : MonoBehaviour  
{  
public AnimationClip clip;  
PlayableGraph playableGraph;  
  
void Start ()  
{

playableGraph = PlayableGraph.CreateGraph();  
var playableOutput = playableGraph.CreateAnimationOutput("Animation", GetComponent<Animator>());​

// Wrap the clip in a playable  
var clipPlayable = playableGraph.CreateAnimationClipPlayable(clip);  
// Connect the Playable to an output  
playableOutput.sourcePlayable = clipPlayable;  
// Plays the Graph.  
playableGraph.Play();  
}  
void OnDisable()  
{

// Destroys all Playables and Outputs created by the graph.  
playableGraph.Destroy();​

}​

}  
  
  
Note that AnimationPlayableUtilities provides a higher level utility functions that simplifies creation and playback of AnimationPlayables.  
  
using UnityEngine;  
  
using UnityEngine.Experimental.Director;  
  
  
[RequireComponent (typeof (Animator))]  
public class PlayAnimation : MonoBehaviour  
{

public AnimationClip clip;  
PlayableGraph playableGraph;  
  
void Start ()  
{

AnimationPlayableUtilities.PlayClip(GetComponent<Animator>(), clip, out playableGraph);​

}  
  
void OnDisable()  
{

// Destroys all Playables and Outputs created by the graph.  
playableGraph.Destroy();​

}​

}  
  
  
  
**Creating an animation blend tree programmatically**  
  
We now introduce the AnimationMixerPlayable, which can blend two or more AnimationClipPlayables, or blend other mixers which themselves blend other clips, etc.  
  
  
The weight of each clip in the blend is adjusted dynamically via SetInputWeight().  
  
  
using UnityEngine;  
using UnityEngine.Experimental.Director;  
  
[RequireComponent (typeof (Animator))]  
public class MixAnimation : MonoBehaviour  
{

public AnimationClip clip1;  
public AnimationClip clip2;  
public float weight;  
  
private PlayableGraph playableGraph;  
private PlayableHandle mixer;  
  
void Start ()  
{

// Creates the graph, the mixer and binds them to the Animator.  
  
mixer = AnimationPlayableUtilities.PlayMixer(GetComponent<Animator>(), 2, out playableGraph);  
// Creates AnimationClipPlayable and connects them to the mixer.  
  
playableGraph.Connect(playableGraph.CreateAnimationClipPlayable(clip1), 0, mixer, 0);  
playableGraph.Connect(playableGraph.CreateAnimationClipPlayable(clip2), 0, mixer, 1);​

}  
  
void Update()  
{

weight = Mathf.Clamp01(weight);  
mixer.SetInputWeight(0, 1.0f-weight);  
mixer.SetInputWeight(1, weight);​

}  
  
void OnDisable()  
{

// Destroys all Playables and Outputs created by the graph.  
playableGraph.Destroy();​

}​

}  
  
  
  
**Blending AnimatorController**  
  
Just like the AnimationClipPlayable wraps an AnimationClip, the AnimatorControllerPlayable can wrap an AnimatorController, allowing us to blend them with other AnimationPlayables.  
  
  
using UnityEngine;  
using UnityEngine.Experimental.Director;  
  
[RequireComponent (typeof (Animator))]  
public class MixController : MonoBehaviour  
{

public AnimationClip clip1;  
public RuntimeAnimatorController controller;  
public float weight;  
  
private PlayableGraph playableGraph;  
private PlayableHandle mixer;  
  
void Start ()  
{

// Creates the graph, the mixer and binds them to the Animator.  
mixer = AnimationPlayableUtilities.PlayMixer(GetComponent<Animator>(), 2, out playableGraph);  
  
// Creates and connects the Playbles to the mixer.  
playableGraph.Connect(playableGraph.CreateAnimationClipPlayable(clip1), 0, mixer, 0);  
playableGraph.Connect(playableGraph.CreateAnimatorControllerPlayable(controller), 0, mixer, 1);​

}  
  
  
void Update()  
{

weight = Mathf.Clamp01(weight);  
mixer.SetInputWeight(0, 1.0f-weight);  
mixer.SetInputWeight(1, weight);​

}  
  
  
void OnDisable()  
{

// Destroys all Playables and Outputs created by the graph.  
playableGraph.Destroy();​

}​

}  
  
  
  
**Controlling the timing of the tree manually**  
  
By default, the Play() method on the PlayableGraph handles all the timing of the tree playback. However, for additional control, it is possible to explicitly set the local time of a Playable For instance, in this example, we pause the playback, and control the time using a variable.  
  
  
using UnityEngine;  
using UnityEngine.Experimental.Director;  
  
[RequireComponent (typeof (Animator))]  
public class PlayWithTimeControl : MonoBehaviour  
{

public AnimationClip clip;  
public float time;  
  
PlayableGraph playableGraph;  
PlayableHandle clipPlayableHandle;  
  
void Start ()  
{

clipPlayableHandle = AnimationPlayableUtilities.PlayClip(GetComponent<Animator>(), clip, out playableGraph);  
// stops time from progressing automatically.  
clipPlayableHandle.playState = PlayState.Paused;​

}  
  
void Update ()  
{

// Control the time manually  
clipPlayableHandle.time = time;​

}  
  
void OnDisable()  
{

// Destroys all Playables and Outputs created by the graph.  
playableGraph.Destroy();​

}​

}  
  
  
Controlling the play state of the tree  
  
  
Similarly, the state of the tree, or a branch of the tree, can be set by changing the Playable.state parameter. The state will propagate to all children of the node, regardless of their previous state. (Keep in mind that if a child node was explicitly paused, setting a parent to Playing state will also set this child to Playing state).  
  
**Creating ScriptPlayable**  
  
The Playable API allows to create custom playables by deriving from ScriptPlayable. By overloading the PrepareFrame method, we can handle the nodes as desired. In this example, the goal is to have animation clips play one after the other. We change the weight of the nodes so that only one clip plays at a time, and we adjust the local time of the clips so that they start at the moment they get activated.  
  
  
Similarly, Custom Playables can also implement the OnSetPlayState and OnSetTime methods, to implement custom behaviours when a playable’s state or local time has changed.

Last edited: **Jan 19, 2017**

[pierrepaul](https://forum.unity.com/members/pierrepaul.112457/), [**Jan 19, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2926897)

[#2](https://forum.unity.com/threads/animationclipplayable.451854/#post-2926897)

[ModLunar](https://forum.unity.com/members/modlunar.1182760/), [nonopblic](https://forum.unity.com/members/nonopblic.1218340/), [Rodolfo-Rubens](https://forum.unity.com/members/rodolfo-rubens.153840/) and [1 other person](https://forum.unity.com/posts/2926897/likes) like this.

1. [**pierrepaul**](https://forum.unity.com/members/pierrepaul.112457/)

[](https://forum.unity.com/members/pierrepaul.112457/)

Unity Technologies

Joined:

Jun 19, 2012

Posts:

[161](https://forum.unity.com/search/member?user_id=112457)

and an example of script playable:  
  
using UnityEngine;  
using UnityEngine.Experimental.Director;  
  
public class PlayQueuePlayable : ScriptPlayable  
{

private int m\_CurrentClipIndex = -1;  
private float m\_TimeToNextClip;  
private PlayableHandle mixer;  
  
public void Initialize(AnimationClip[] clipsToPlay)  
{

handle.inputCount = 1;  
mixer = handle.graph.CreateAnimationMixerPlayable(1);  
handle.graph.Connect(mixer,0, handle,0);  
mixer.inputCount = clipsToPlay.Length;  
for (int clipIndex = 0 ; clipIndex < mixer.inputCount ; ++clipIndex)  
{

handle.graph.Connect(handle.graph.CreateAnimationClipPlayable(clipsToPlay[clipIndex]),0, mixer, clipIndex);​

}​

}  
  
override public void PrepareFrame(FrameData info)  
{

// Advance to next clip if necessary  
m\_TimeToNextClip -= (float)info.deltaTime;  
if (m\_TimeToNextClip <= 0.0f)  
{

m\_CurrentClipIndex++;  
if (m\_CurrentClipIndex < mixer.inputCount)  
{

var currentClip = mixer.GetInput(m\_CurrentClipIndex).GetObject<AnimationClipPlayable>();  
// Reset the time so that the next clip starts at the correct position  
currentClip.handle.time = 0;  
m\_TimeToNextClip = currentClip.clip.length;​

}​

}​

// Adjust the weight of the inputs  
for (int clipIndex = 0 ; clipIndex < mixer.inputCount ; ++clipIndex)  
{

if (clipIndex == m\_CurrentClipIndex)

mixer.SetInputWeight(clipIndex, 1.0f);​

else

mixer.SetInputWeight(clipIndex, 0.0f);​

}​

}​

}  
  
[RequireComponent (typeof (Animator))]  
public class PlayQueue : MonoBehaviour  
{

public AnimationClip[] clipsToPlay;  
PlayableGraph playableGraph;  
void Start ()  
{

playableGraph = PlayableGraph.CreateGraph();  
var playQueue = playableGraph.CreateScriptPlayable<PlayQueuePlayable>().GetObject<PlayQueuePlayable>();  
playQueue.Initialize(clipsToPlay);  
  
var playableOutput = playableGraph.CreateAnimationOutput("Animation", GetComponent<Animator>());  
playableOutput.sourcePlayable = playQueue;  
playableGraph.Play();​

}  
  
void OnDisable()  
{

// Destroys all Playables and Outputs created by the graph.  
playableGraph.Destroy();​

}​

}

[pierrepaul](https://forum.unity.com/members/pierrepaul.112457/), [**Jan 19, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2926913)

[#3](https://forum.unity.com/threads/animationclipplayable.451854/#post-2926913)

[Peter77](https://forum.unity.com/members/peter77.308146/) and [ArthurT](https://forum.unity.com/members/arthurt.709908/) like this.

1. [**PhilSA**](https://forum.unity.com/members/philsa.343471/)

[](https://forum.unity.com/members/philsa.343471/)

Joined:

Jul 11, 2013

Posts:

[1,056](https://forum.unity.com/search/member?user_id=343471)

@[pierrepaul](https://forum.unity3d.com/members/pierrepaul.112457/)  
I see you also got rid of the GC allocs when calling SetInputWeight()! Great job!

[PhilSA](https://forum.unity.com/members/philsa.343471/), [**Jan 19, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2927056)

[#4](https://forum.unity.com/threads/animationclipplayable.451854/#post-2927056)

1. [**PhilSA**](https://forum.unity.com/members/philsa.343471/)

[](https://forum.unity.com/members/philsa.343471/)

Joined:

Jul 11, 2013

Posts:

[1,056](https://forum.unity.com/search/member?user_id=343471)

By the way, is there a planned way to apply AvatarMasks to certain Playables?  
  
I tried to accomplish this by creating an empty AnimatorController with the mask I want on its base layer, and play some AnimClipPlayables on that AnimationController, but it didn't seem to work

[PhilSA](https://forum.unity.com/members/philsa.343471/), [**Jan 23, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2931792)

[#5](https://forum.unity.com/threads/animationclipplayable.451854/#post-2931792)

1. [**pierrepaul**](https://forum.unity.com/members/pierrepaul.112457/)

[](https://forum.unity.com/members/pierrepaul.112457/)

Unity Technologies

Joined:

Jun 19, 2012

Posts:

[161](https://forum.unity.com/search/member?user_id=112457)

PhilSA said: [↑](https://forum.unity.com/goto/post?id=2931792#post-2931792)

*By the way, is there a planned way to apply AvatarMasks to certain Playables?  
  
I tried to accomplish this by creating an empty AnimatorController with the mask I want on its base layer, and play some AnimClipPlayables on that AnimationController, but it didn't seem to work*

Hi ! we are working on it ! We will publish the AnimationLayerMixerPlayable which allows to specify a mask.

[pierrepaul](https://forum.unity.com/members/pierrepaul.112457/), [**Feb 1, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2943278)

[#6](https://forum.unity.com/threads/animationclipplayable.451854/#post-2943278)

[PhilSA](https://forum.unity.com/members/philsa.343471/) likes this.

1. [**PhilSA**](https://forum.unity.com/members/philsa.343471/)

[](https://forum.unity.com/members/philsa.343471/)

Joined:

Jul 11, 2013

Posts:

[1,056](https://forum.unity.com/search/member?user_id=343471)

pierrepaul said: [↑](https://forum.unity.com/goto/post?id=2943278#post-2943278)

*Hi ! we are working on it ! We will publish the AnimationLayerMixerPlayable which allows to specify a mask.*

In 5.6b6, I see that AnimationLayerMixerPlayable was added, but I'm not sure how to add one to a PlayableGraph.   
  
I'm basically looking for a PlayableGraph.CreateAnimationLayerMixerPlayable() but it's not there. Is there another way right now to add an AnimationLayerMixerPlayable to a graph?

[PhilSA](https://forum.unity.com/members/philsa.343471/), [**Feb 1, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2943575)

[#7](https://forum.unity.com/threads/animationclipplayable.451854/#post-2943575)

1. [**KreolDev**](https://forum.unity.com/members/kreoldev.725018/)

[](https://forum.unity.com/members/kreoldev.725018/)

Joined:

Nov 23, 2014

Posts:

[4](https://forum.unity.com/search/member?user_id=725018)

Hi. I've been trying to build a small animation system with new playables for the last 2 days, but it either behaves weirdly or crashes the editor entirely.  
  
What I'm trying to achieve is to have a graph in which I insert animations as neccessary (for example, an object can contain an animation on how to use it and I don't want to put it in every graph of every NPC, but rather to dynamically add it and crossfade from controller to it and back).  
  
I've tried the animation mixer, and added a controller and a clip to it (or two controllers, or two clips, or a clip and two controllers, you name it).   
  
Code: <http://pastebin.com/ZiDyQ6gP>

Spoiler: Results

What happens is that when an NPC comes close enough to the sphere an interaction is enacted, which asks the component above to play a clip of "use" animation, which starts with the weight of 0. Unfortunately for some reason when I add it to the mixer it forces an object to jump back to its origin.  
  
I've also tried this code, using the generic mixer and creating everything manually. Unfortunately it crashes if I connect a playable to it after I've asked it to start playing.  
<http://pastebin.com/FSjYFeVp>  
  
  
Any insights?

[KreolDev](https://forum.unity.com/members/kreoldev.725018/), [**Mar 10, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2988386)

[#8](https://forum.unity.com/threads/animationclipplayable.451854/#post-2988386)

1. [**PhilSA**](https://forum.unity.com/members/philsa.343471/)

[](https://forum.unity.com/members/philsa.343471/)

Joined:

Jul 11, 2013

Posts:

[1,056](https://forum.unity.com/search/member?user_id=343471)

[@KreolDev](https://forum.unity.com/members/725018/)  
I'd like to see a dev comment on this, but I'm not sure playables are meant to be used like that.  
  
I have a situation that is somewhat similar to yours: a player can pickup skills, and each skill contains its AnimationClip. What I do is that instead of modifying the playables graph on the fly whenever a skill is picked up, I generate the entire graph with all the possible skill anims on start (there is some sort of manager somewhere that knows of every possible skill in the game). This way you don't have to append stuff to the graph while it's running and possibly cause weird behaviour.  
  
However, I understand the appeal of using your approach, and this is what I originally wanted to do before I noticed that it caused problems. So if a dev knows of a way to do this, I'd like to know  
  
Edit: maybe try Playablegraph.Stop(), then add the new playable, then Playablegraph.Play() and see if it fixes the problem? It might reset all the timings and/or weights so you'll have to store the data of the current state somewhere and reapply it on Play()

Last edited: **Mar 10, 2017**

[PhilSA](https://forum.unity.com/members/philsa.343471/), [**Mar 10, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2988489)

[#9](https://forum.unity.com/threads/animationclipplayable.451854/#post-2988489)

[theANMATOR2b](https://forum.unity.com/members/theanmator2b.655602/) likes this.

1. [**DavidGeoffroy**](https://forum.unity.com/members/davidgeoffroy.685159/)

[](https://forum.unity.com/members/davidgeoffroy.685159/)

Unity Technologies

Joined:

Sep 9, 2014

Posts:

[282](https://forum.unity.com/search/member?user_id=685159)

[@KreolDev](https://forum.unity.com/members/725018/):  
If you haven't already done so, please file a bug with the case where it crashes. It's never normal for Unity to crash.   
I think you're supposed to use PlayableGraph.DestroyPlayable, not Destroy directly though (wild guess).  
  
As for the jump, it's hard to tell at a glance. I get the feeling that something you're doing is forcing a reset of the graph. I'm not sure exactly what, but you can feel free to file a bug with it; we'll look at it.  
  
[@PhilSA](https://forum.unity.com/members/343471/):   
Your approach should be more efficient. Any time new clips are added, we need to recalculate the bindings (the total list of everything the Animator writes to), and this is costly. The cost scales on the number of different curves you write to, and the number of clips you have. This cost already exists in every Animation solution in Unity (Humanoid, Generic, Legacy), it's not specific to Playables.   
  
Also, If you connect clips that write to new curves mid-play, you'll also save new default values, that might or might not be what you want.  
  
By pre-connecting the clips, you won't have to pay this cost every time you need to change clips, and you'll make sure to get the default values from scene load/instantiation.

[DavidGeoffroy](https://forum.unity.com/members/davidgeoffroy.685159/), [**Mar 13, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2991773)

[#10](https://forum.unity.com/threads/animationclipplayable.451854/#post-2991773)

[theANMATOR2b](https://forum.unity.com/members/theanmator2b.655602/) likes this.

1. [**PhilSA**](https://forum.unity.com/members/philsa.343471/)

[](https://forum.unity.com/members/philsa.343471/)

Joined:

Jul 11, 2013

Posts:

[1,056](https://forum.unity.com/search/member?user_id=343471)

DavidGeoffroy said: [↑](https://forum.unity.com/goto/post?id=2991773#post-2991773)

*...*

Thanks for the reply. I'll definitely make some tests to try modifying a graph "at runtime" to see what kind of performance impact it creates, and how it affects resetting the graph and such.  
  
While you're here, I'd like to ask: is there another way to create a playable in a graph, aside from the PlayableGraph.CreateXPlayable() methods? I'm asking because I can see that AnimationLayerMixerPlayable has made it to the beta releases, but I see no way of creating them in a graph  
  
For example, would this work?

Code (CSharp):

* 2. \_baseMixer = AnimationPlayableUtilities.PlayMixer(Animator, 1, **out** \_playableGraph);
  3. AnimationLayerMixerPlayable animLayerMixerPlayable = [new](http://www.google.com/search?q=new+msdn.microsoft.com) AnimationLayerMixerPlayable();
  4. \_playableGraph.Connect(animLayerMixerPlayable, 0, \_baseMixer, 0);

Last edited: **Mar 13, 2017**

[PhilSA](https://forum.unity.com/members/philsa.343471/), [**Mar 13, 2017**](https://forum.unity.com/threads/animationclipplayable.451854/#post-2991864)

[#11](https://forum.unity.com/threads/animationclipplayable.451854/#post-2991864)

1. [**DavidGeoffroy**](https://forum.unity.com/members/davidgeoffroy.685159/)

[](https://forum.unity.com/members/davidgeoffroy.685159/)

Unity Technologies

Joined:

Sep 9, 2014

Posts:

[282](https://forum.unity.com/search/member?user_id=685159)

PhilSA said: [↑](https://forum.unity.com/goto/post?id=2991864#post-2991864)

*While you're here, I'd like to ask: is there another way to create a playable in a graph, aside from the PlayableGraph.CreateXPlayable() methods? I'm asking because I can see that AnimationLayerMixerPlayable has made it to the beta releases, but I see no way of creating them in a graph*

No, this is the only creation method.  
  
AnimationLayerMixerPlayable still needs some other structures that are Editor only to be exposed in order to work properly in C#-land. I'm not sure why it got exposed in 5.6, maybe so the graph visualizer can show it, even if it's not really interactable.  
  
It will be fully exposed in 2017.1.

窗体底端