Playable Graph is part of the Playables API in Unity. The Playables API is a system that allows you to create, manipulate, and control various types of data-driven objects. It is used for tasks such as animation, audio, and video processing. It provides ways to create and manipulate playables and playables graphs, which are structures that define the relationships and interactions between playables.

The Playable Graph, specifically, is a container for playables and playable outputs. It allows you to organize and control the flow of data between different playables, creating complex behaviors and interactions. Playable Graphs are commonly used in animation systems to control character animations, blend animations, and create dynamic sequences.

It is a system for creating and managing interactions within your project. It's part of Unity's animation and timeline system, and it enables you to create dynamic and interactive experiences.

“The PlayableGraph defines a set of playable outputs that are bound to a GameObject or component The PlayableGraph also defines a set of playables and their relationships. The PlayableGraph is responsible for the life cycle of its playables and their outputs. Use the PlayableGraph to create, connect, and destroy playables.”[1]

Playable Graphs is like the conductor of an orchestra, coordinating different instruments (animations, audio, etc.).

What are playables? They are low-level, data-driven object that represents an element of animation, audio, or other time-based interactions.

What are playable outputs?

For animations: Represents an animation clip or a blend tree of animation clips. It is used for controlling character animations, blending between animations, and creating complex animation behaviors.

For audio: Represents an output for audio playables. It connects the audio playable to an audio source, allowing the audio to be played.

[1] From Unity documentation

1. Create an AnimationClip:
    a. In Unity, create or import an AnimationClip named, for example, "RunAnimation." This will be the animation you want to play.
2. Write a Script:
    a. Create a new C# script in your Unity project. Name it something like "PlayGraph.cs". This script will handle the animation playback.
3. Open the Script:
    a. Open the "PlayGraph.cs" script in your preferred code editor (e.g., Visual Studio).
4. Declare Variables:
    a. Declare variables to store the PlayableGraph, AnimationClipPlayable, and the AnimationClip you want to play.
5. Initialize PlayableGraph:
    a. In the Start() method, create a PlayableGraph and set its time update mode to GameTime.
6. Create AnimationClipPlayable:
    a. Create an AnimationClipPlayable for your animation using the AnimationClip you declared.
7. Create AnimationPlayableOutput:
    a. Create an AnimationPlayableOutput and connect it to the Animator component on your GameObject.
8. Play the Graph:
    a. Use playableGraph.Play() to start the animation playback.
9. Clean Up:
    a. In the OnDestroy() method, destroy the PlayableGraph to release resources when the object is destroyed.
10. Make sure to be able to visualize the graph:
    a. Download or clone the the visualizer from: https://bitbucket.org/Unity-Technologies/playablegraphvisualizer
    b. On your void start() or on your void update() make sure to have: GraphVisualizerClient.Show(myPlayable);

c. In unity, you can access the visualizer by going into window -> analysis -> PlayableGraph Visualizer

11. Attach Script to GameObject:
    a. Attach the "PlayGraph" script to the GameObject that has the Animator component.

Resources:

https://assetstore.unity.com/packages/3d/characters/humanoids/banana-man-196830

https://www.mixamo.com/#/

https://docs.unity3d.com/Manual/Playables-Graph.html

https://dev.rbcafe.com/unity/unity-5.3.3/en/Manual/Playables-GraphVisualizer.html