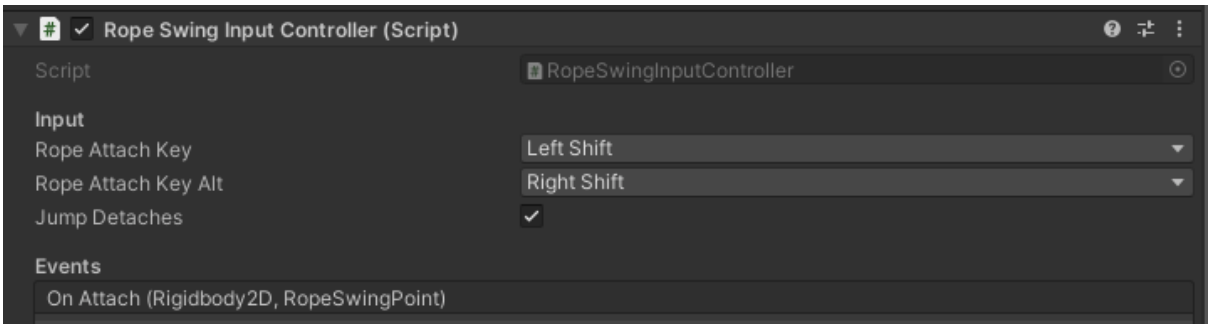
Simple 2D Rope Swing

Documentation

Getting Started

Input

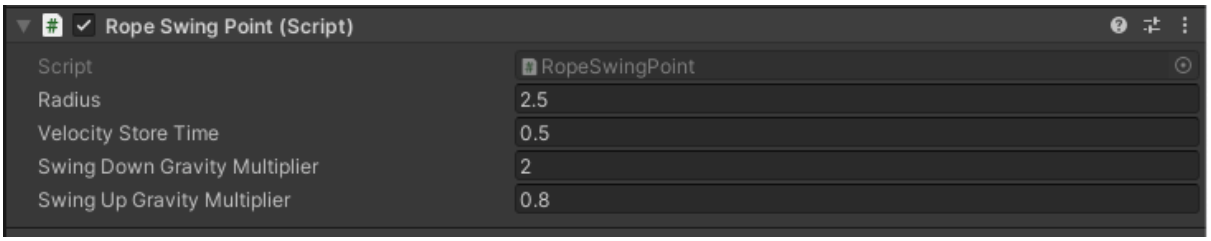
Your player will need the “Rope Swing Input Controller” component:



This will require a Rigidbody2D, if one isn’t already on the object it will be added. You can remove this requirement by deleting the [RequireComponent] attribute at the top of the script, and give it a reference to some other Rigidbody2D if required.

Swing points

Each point you want to be able to swing off will need the “Rope Swing Point” component.



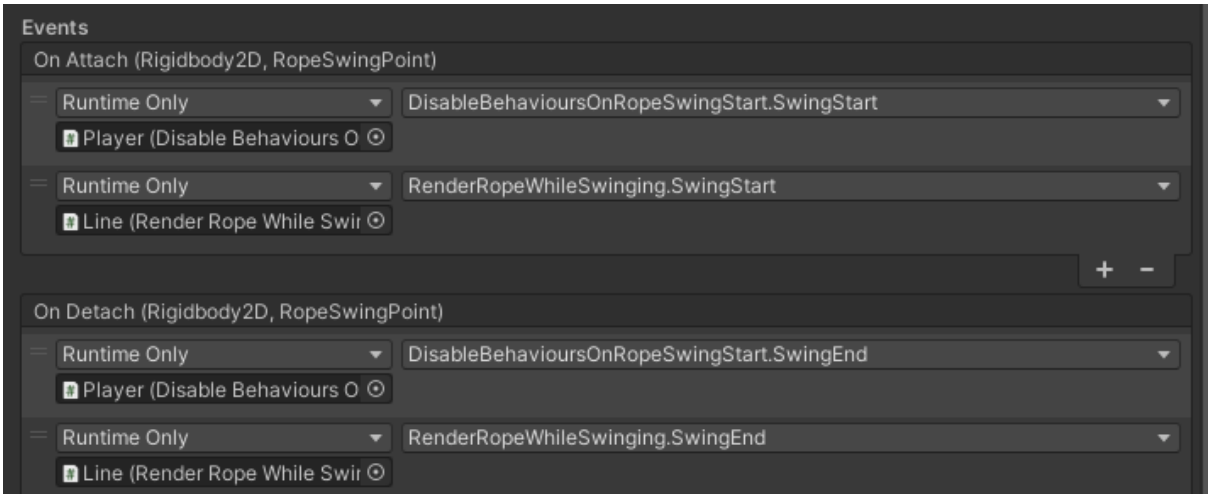
And that’s it! With these two components, your character will swing and detach. For more detail about each component and some scripting information, continue reading.

Rope Swing Input Controller

This object controls the input for the rope swing. The “Jump Detaches” bool controls whether or not pressing the jump key allows the player to detach from the rope. If it is false, the player can only detach if the “attach” key is pressed while swinging.

Events

There are two events: OnAttach and OnDetach. These are called when a rope is first attached and when a rope is detached. **Use these to extend the functionality** - you shouldn’t need to change the original code unless there’s something very specific you want to do. I’ve included examples for disabling monobehaviours while swinging, and rendering a line with a line renderer while swinging, I suggest you take a look at those and the demo scene to figure out how it works. They are assigned in the Unity inspector:



For convenience, I’ve also included C# events that have the same functionality, use whichever you like - C# events are a little faster and are a lot more dynamic, with the ability to easily subscribe to and unsubscribe from. Unity events are assigned in the inspector and are generally a little easier to work with, and you don’t have to remember to unsubscribe from them!

Legacy Input System

If you're using the old input system, it will use the keys assigned on the component. There is also a RebindKey function to allow runtime control remapping. If the "Jump Detaches" bool is true, it uses Input.GetButtonDown("Jump") as the "Jump" input.

New Input System

If you're using the new input system, there are two public functions: OnAttachInput and OnJumpInput. You should assign these in your PlayerInput component, or bind them to your InputActions in OnEnable/OnDisable. If you're not already familiar with the PlayerInput component, it makes using the new input system MUCH easier! All you've got to do is change the "behaviour" to "Invoke Unity Events" and you'll have a much better time. Have a look here: https://docs.unity3d.com/Packages/com.unity.inputsystem@1.0/manual/Components.html

Rope Swing Point

This is where the magic happens!

The radius field describes the activation range for the swing point and changes the gizmo. If the player is inside the white circle, they'll be able to swing from that rope swing point.

Storing velocity

The velocity store time denotes how long the max velocity will be kept for - generally, you want to cheat in favour of the player. This is similar to jump input buffering and coyote time, and widens the window in which the player can perform a "perfect" swing and gain the maximum amount of velocity.

Momentum

The swing down/swing up gravity modifiers alter the rigidbody's gravity scale. Increasing the gravity multiplier on swing down will cause the player to gain more momentum from a swing, and decreasing it on swing up will reduce the amount of momentum lost while swinging up/away from gravity.

Starting the swing

The default behaviour of the rope swing uses the player's position relative to the swing point to decide which way to start the swing. This is done in the ImmediateChangeVelocityToTangent function on line 99, this is probably the most likely place you might want to edit the actual code of this plugin.