Spline Kit Pro User Manual

Overview

Spline Kit Pro is a spline animation toolkit. It allows you to build custom splines in Unity's editor and animate your own Game Objects along them. With this toolkit you'll be able to add trigger events along the spline that allow you to script things such as animation changes, spawning particle effects, changing the speed and acceleration, playing sounds, and more, without writing any code. Additionally, these trigger events can call into your own scripts, triggering your own logic with support for passing multiple parameters, including your own game or data object.
New Features

Features

- Custom spline creation, including loops.
- Standard and Sculpt editor modes for easy spline editing.
- Event Triggers - Trigger particles, sounds, animations, and your own custom scripts, anywhere along the spline.
- Custom Rotation and Scaling along the whole spline, or, segments you define; with seamless blending between rotations and scaling.
- Branch a spline into any number of sub splines and join them back anywhere (on the branching spline, or any other spline you chose). You can use one of the built in methods for choosing a branch to take, or, implement your own logic.
- Join any spline to any other spline, anywhere you choose.
- Use animation curves to control the speed of your GameObject.
- Lob generator. Dynamically generate lob paths at runtime to have GameObjects hit specified targets.

Version History

- **Version 1.0.0** [06/23/2016]
    - First release candidate.

- **Version 1.0.1** [07/25/2016]
    - Removed SplineKitPro from the main menu bar.

- **Version 1.0.2** [08/03/2016]
    - First release made available.
    - Resubmitted per request, no changes.

- **Version 1.0.3** [08/09/2016]
    - Updated blend nodes to use the spline rotation/scale when blending in or out of the node, rather than potentially competing with another blend node.
    - Disabled the Spline Edit Plane on start so that it doesn't inadvertently collide with game objects.
    - Added an option to rotation blend nodes to disable banking while under the influence of the node.
    - Changed the rotation blend node editor to only recalculate the rotation from the euler inputs if it had changed.
    - Change the spline animator to keep evaluating when it reaches the end of the spline. This may be needed if you have some external script or animation moving the spline and want the animated game object to remain on the spline while stopped at the end.
    - Fixed spline corruption when attempting to insert a point before the first or after the last.
- **Version 1.0.4** [08/23/2016]
    - Fixed an issue where dynamic generated lob paths would not generate outside of the editor.
    - Added a function to clear delayed triggered events on the spline animator.
    - Made the furthest rotation node with influence be the one evaluated (rather than the first). This should correctly resolve conflicts in overlapping rotation nodes.
    - Added the ability to delay generated lob paths.
    - Added a check to delayed commands to ensure the animator hadn't already been destroyed.

- **Version 1.1.1 Beta** [09/08/2016]
    - Added the ability to the lob generator to pre-generate the lob path.
    - Added in game renderer for splines.
    - Send out spline complete messages when the animator is destroyed before reaching the end of the spline.
    - Added options to re-generate the spline each update. This is useful if your spline is moving.
    - Added options to lob generator to remove spline and/or animator on complete.
    - Separated lob generation from lob animation.
    - Updated Lob Generator to pre-generate paths for launching multiple objects.

- **Version 1.1.2 Beta** [11/02/2016]
    - Allow spline animator t val scrubbing when speed is 0.
    - Added the ability to set the animation layer when using animation for the motion.
    - Fixed a bug when using animation motion with a negative speed.
    - Fixed potential floating point error when splines are very small.
    - Disabled shadows for spline rendering.

- **Version 1.2.0 Beta rc3** [01/20/2017]
  Added Features:
    - Added storing off nodes via distance, so they don't move when modifying the spline.
    - Added chaining of rotation and scale blend nodes.
    - Added control point keyframe rotation.
    - Added parameter to adjust the spline draw resolution.
    - Added IndexOf function to spline API.
    - Optimized SKRandomSequence to not create a new list on every reshuffle.
    - Changed SKRandomSequence to match Unity's convention of having the max be exclusive.

  Bug fixes:
    - Fixed the incorrect reporting of the first point tvalue when looped vs non-looped.
    - Fixed an editor bug where minus button for omitters/receivers could be pushed offscreen
    - Fixed a Unity 5.5 warning.
    - Fixed a bug in SKRandomSequence where 1 element of the sequence may be missed.
    - Fixed branch random no repeat selection to include the main spline.
    -

- **Version 1.2.0 Beta rc4** [03/04/2017]
  Added Features:
    - Added a Vector3, and 2 Gameobject parameters to the SKCmd structure for calling custom functions.
    - Added a semi-transparent material for the spline edit plane.
    - Added an option to allow trigger nodes to be triggered by collision.
    - Added tool tips to spline triggers to identify which parameters in the SKCmd structure are being used.
    - Fixed "Smoother" label width so it doesn't get cut off in the inspector.
    - Added a count variable to SKRandomSequence.

  Bug fixes:
    - Fixed point nodes having a t value of 0 when the spline was not looped.

- **Version 1.2.0 Beta rc5** [03/04/2017]
  Bug fixes:
    - Fixed nested branch and trigger nodes being triggered when animator is on a parent spline.

Creating Splines (Alt-Shift-S)

To create a new spline game object, use the menu ‘GameObject->Spline Kit Pro->Create New Spline’ or In the hierarchy view ‘Create->Spline Kit Pro->Create New Spline’ or use the shortcut Alt-Shift-S

Scene Editing

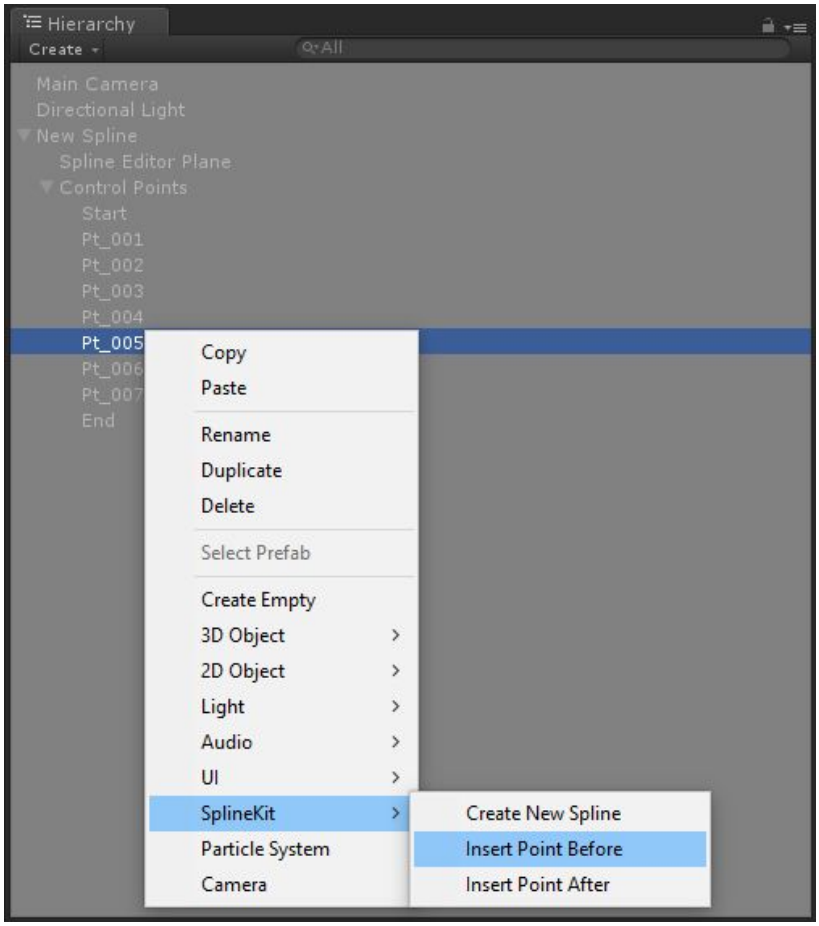
Once you have your spline created, you'll probably want to add some points or edit it.

Adding Control Points (*Shift+LMB*):

With your spline selected in the hierarchy, use 'Shift + Left Mouse Button' in the scene view to place a point. The point will be placed at the current mouse position at depth according to your *Point Depth Priority*.

Inserting Control Points:

Select a control point (either via the scene view, or from the hierarchy view). Then right click on the selected point in the hierarchy to bring up the context menu and select '*Spline Kit Pro->Insert Point Before*' or '*Spline Kit Pro->Insert Point After*'. This will insert a point halfway between the selected point and the previous (or next) point.

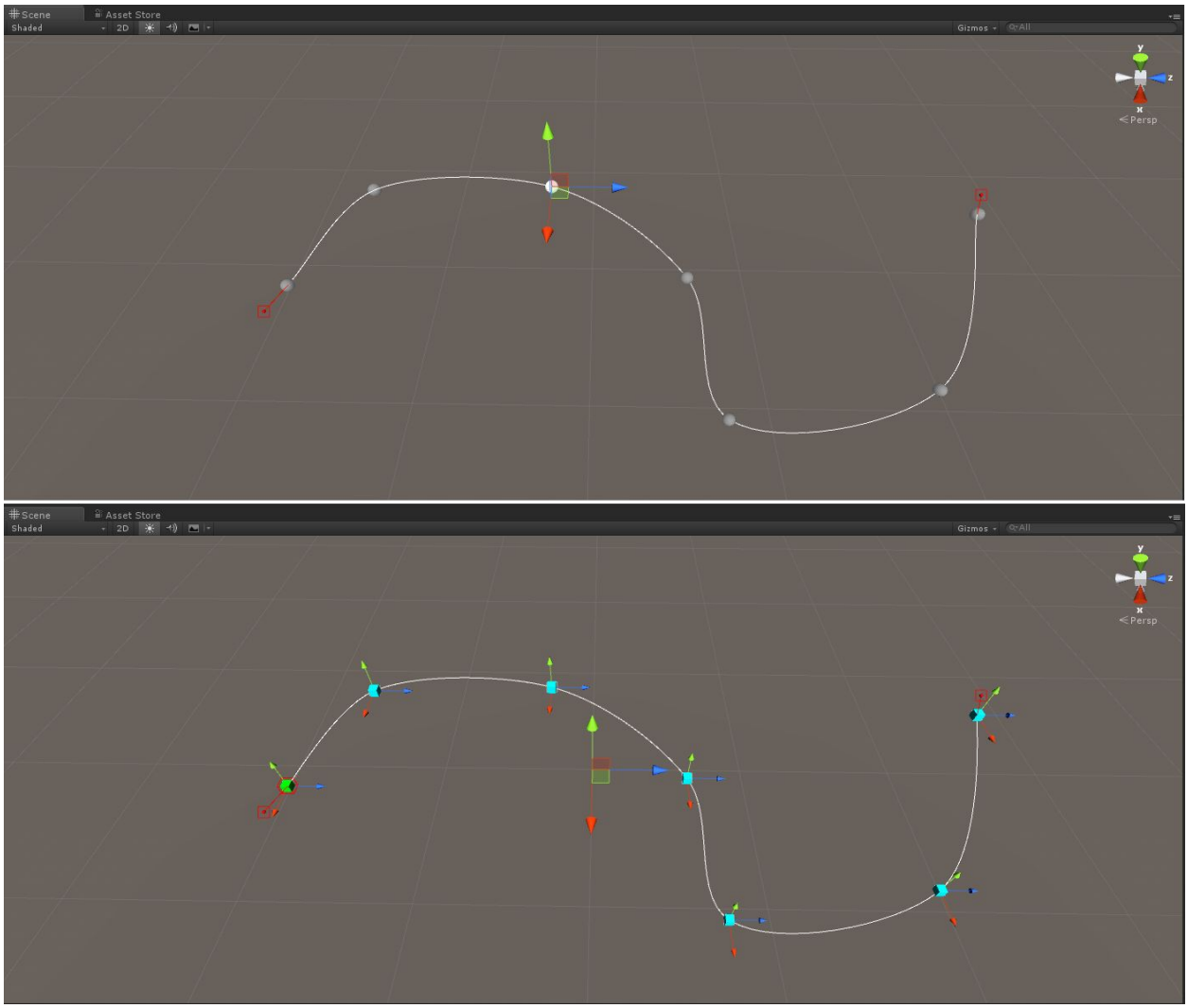


Removing Control Points:

Simply select the control point and hit ‘delete’.

Editing Control Points:

There are two modes for editing control points, ‘*Standard*’ and ‘*Sculpt*’ (*See* *Spline Inspector / Edit / Edit Mode*). *Standard* (top) lets you edit points just like any other Unity scene object, select the point and use the transform tools to move it. *Sculpt* mode (bottom) allows you edit all points without having to select the point each time.



Spline Inspector

Info

Version	The version of Spline Kit Pro that the spline was created with.
Length	This is the current length of your spline.

Render

Always Render	Renders the spline even when it is not selected. The color box allows you to set the color of the spline. (Editor Only)
Spline Color	The color of the spline.
Point Handle Size	The size of the transform tool for points (Sculpt mode only).
Rotation Handle Size	The size of the rotation handles on rotation blend nodes.
Scale Handle Size	The size of the scale handles on scale blend nodes.

Setup

Parameterization	The parameterization the spline uses when evaluating (Centripetal is used by default, but you can use Uniform or Chordal if you prefer). This is an advanced feature which you can most likely ignore.
Spline Up	This is considered up for objects moving along the spline. In other words, the top of your objects will point in this direction (if not being modified by any other rotation factors; such as, rotation blend nodes, or any animations your object might be doing).
Auto Manage End Points	Automatically positions the start and end control points (these are automatically generated and control the curvature at the ends of the spline). Spline Kit Pro can automatically manages these for you, but if you want or need to control them, just disable this.

Edit

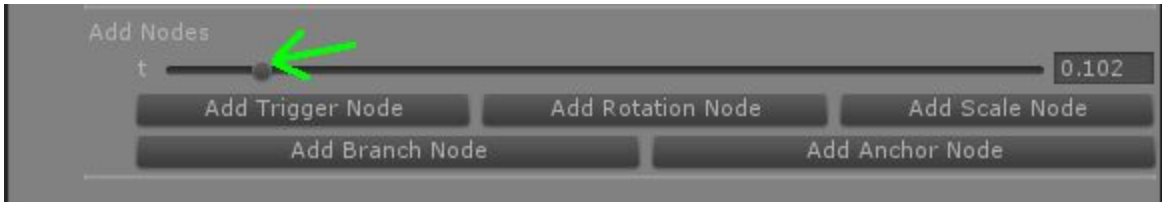
Edit Mode	● ***Standard*** - Edit control points as standard Unity objects. ● ***Sculpt*** - Draws transforms on all points at once allowing for easy point manipulation.
Point Depth Priority	When you adding a point to a spline, Spline Kit Pro needs to determine at what depth the point should be added. It uses three methods to determine this and this field allows you to set which method should be used first, second, and third. The three methods are ● ***Collision*** - Spline Kit Pro will project a ray from the camera through the mouse point into the scene and use the first collision intersection it finds. ● ***Spline Plane*** - Each newly created spline has a child *'Spline Editor Plane'* which can be used for placing points (You can enable the visibility of this plane by selecting it and toggling the *'Show Spline Plane'* option). Again, Spline Kit Pro will cast a ray looking for the first collision with it's own spline plane. ● ***Pivot*** - This will use distance from the camera to its pivot and project that distance along the projected ray through the mouse point.
Freeze Trans	Locks the translation along the specified axis.
Freeze Rotation	Locks the rotation along the specified axis.
Freeze Scale	Locks the scale along the specified axis.
Flatten	Flattens the spline onto the specified plane with the specified height. If 'Include Branches' is checked it will also flatten any branches belonging to the spline.
Add Nodes	Add nodes to the spline. Use the slider to position the placement box along the spline, then use the buttons to add the nodes you require. If ' Automatically Select Node When Added' is selected the newly created node will automatically be selected in the hierarchy (except for Anchor nodes).

Misc Buttons

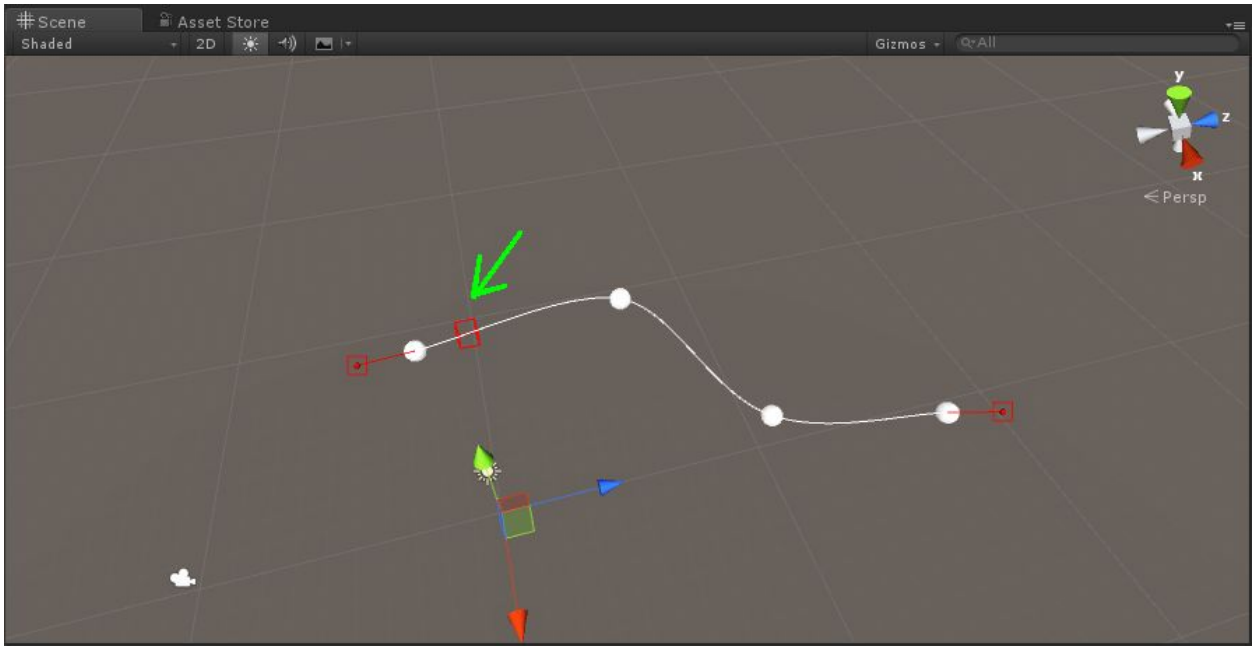
Make Loop	Creates a loop, welding the last point to the first.
Center Origin	Recalculate the center of the spline by calculating the center of the control points.
Reverse Spline (Leave Triggers)	Reverses the direction of the spline, leaving the triggers as they are.
Reverse Spline (Reverse Triggers)	Reverses the direction of the spline, and the triggers.
Clear	Removes all points from the spline.
Recache	Forces recaching of the internal points. Use if your spline has become corrupted.

Nodes

Nodes are placed along the spline to give the spline certain properties or trigger actions or events (such as, triggering an animation, playing a sound, firing a particle, or calling your own custom routine). Additionally, some nodes can have a direction, such that it's only recognized when encountered from a certain direction; e.g., from forward motion, reverse motion, or both. To add a node to a spline, drag the *'t'* slider under 'Add Nodes' on the spline inspector to position the where you'd like to place the node.



This will move the node placement box along the spline.



Once in position, hit the add button of the type of node you'd like to add. There are 5 types of nodes, *'Trigger Node'*, *'Rotation Node'*, *'Scale Node'*, *'Branch Node'*, and *'Anchor Node'*. *(See the following sections for details of how to use each)*.

Triggering Actions and Events (Trigger Nodes)

Trigger nodes are used to trigger actions or events. They are placed along the spline (using the Spline Inspector) and when your object passes through the trigger it will do the specified actions. It can be used to trigger one (or more) of the predefined actions, and/or, execute your own logic. Additionally, you can choose which GameObjects may trigger the event and which can receive the event *(See 'Receiver and Omitter Lists' below).*

Inspector

Direction	Both, Forward, Backward. Specifies the direction for which the trigger is active. For example, if Forward is specified, the trigger will only be executed if your GameObject is traveling forward along the spline (from t=0 to t=1).
T Value	The T value at which the node is placed. This is a normalized value (0.0 - 1.0) along the spline; such that a T value of 0.0 is located at the start of the spline, and a T value of 1.0 is located at the end of the spline.
Commands	A list of commands to be executed when triggered. *See Commands below.*

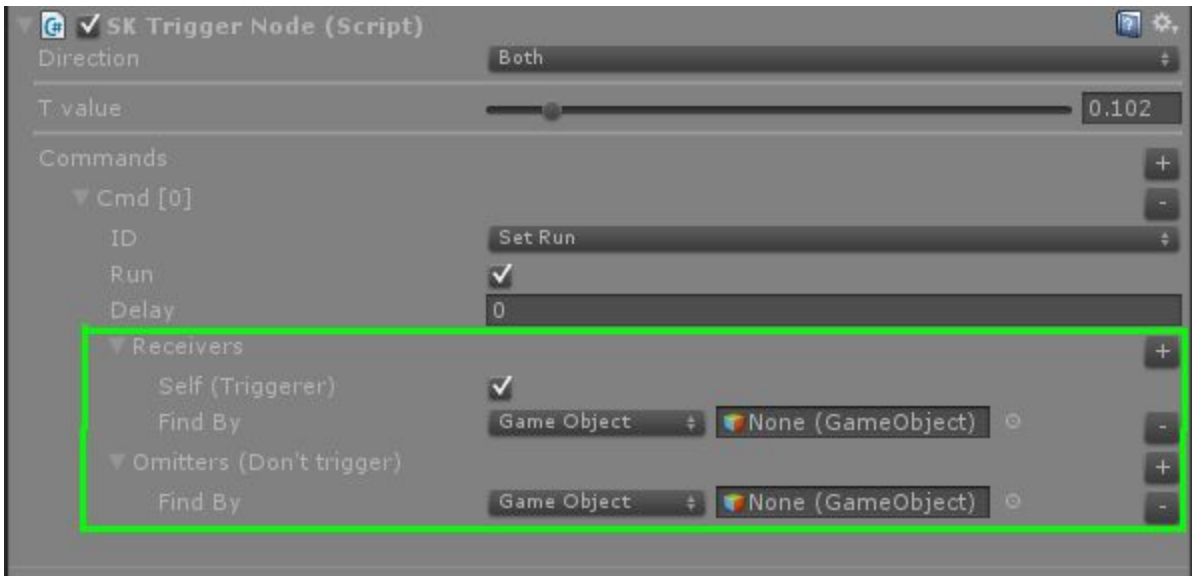
Commands

Set Active	Sets a GameObject active or inactive.
Set Run	Pauses or resumes a GameObject animating along the spline.
Set Speed	Sets the speed and acceleration (if enabled) of a GameObject animating along the spline.
Set Anim Trigger	Sets an animation trigger defined in a GameObjects Animation Controller *(see Unity Animator)*. This is can be used to trigger a specific animation.
Reset Anim Trigger	Resets the animation trigger. Normally you would never need to use this, but it can be useful if your animation controller contains transition logic that requires it.
Play Sound	Plays the sound specified by the clip (path relative to the Resources folder). Note: This is intended for very simple audio, it

	works by checking the GameObject for an audio source with the clip already loaded, if it finds one, it plays it; otherwise, it adds a new AudioSource to the GameObject with the specified clip and plays it.
Play Audio Source	Plays the specified AudioSource. This is a bit more useful than Play Sound with more complicated audio. Here you maintain all the audio sources and settings, and this will simply call play on it.
Instantiate Prefab	Instantiates the specified prefab. If a locator transform is specified, it will instantiate the prefab with the given transform's position and rotation. You can also specify a 'Destroy After Delay', after which, it will automatically be destroyed (leave at 0 if you do not want it to be destroyed).
Set Anim State Time	This allows you to set the current animation playing and the specific frame (via normalized time) on a GameObject. ● *State* - The state in the Animation Controller that has the animation you want to play. ● *Normalized Time* - The normalized time (0.0 - 1.0) in the animation you want to start at.
Apply Motion Curve	This is an advanced feature that allows you to apply the motion from an animation curve to the speed of a GameObject along the spline. To do this, you'll need to have a curve in a Unity animation that you want to use, and then use 'Spline Kit Pro->Animation Curve Extractor' to extract it for use by Spline Kit Pro. *(See Extracting Animation Curves for Motion for more details).*
Execute Function	Calls the specified function. The MonoBehaviour containing the function must be added to the GameObject. The function can take a *SKCmd* as a parameter that will contain all the arguments.

Receiver and Omitter lists

Below each command are two lists: *'Receivers'* and *'Omitters'.* Receivers define who can receive the event, and Omitters define who can not trigger the event.



By default, all GameObjects animating on the spline will receive an event triggered on the spline, and all will trigger the event. To add a receiver or omitter, just click the '+' button to add a new entry. You then have 3 options to specify the GameObject:

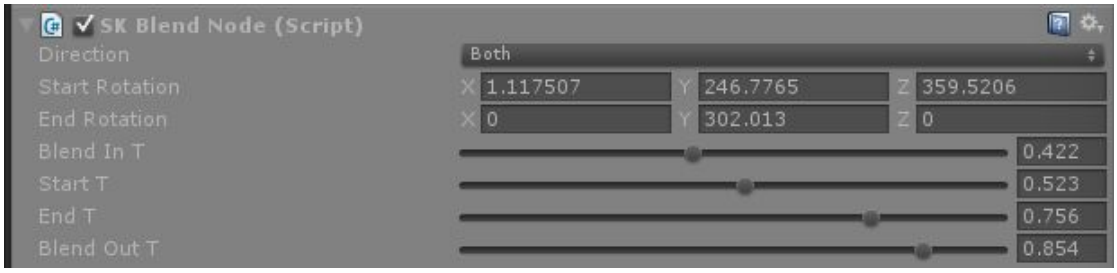
- ***Game Object*** - Scene reference, just drag the GameObject from the scene hierarchy into the field.
- ***Name*** - This will lookup the GameObject by name at runtime (where name is the name of the GameObject in the scene at runtime). This is useful if the GameObject is added dynamically at runtime and isn't part of the scene.
- ***Tag*** - This will lookup all GameObjects with the given tag at runtime. Again, useful if they aren't part of the scene or if you need a group to receive the event.

You may have noticed *'Receivers'* have 1 entry by default called '*Self (Triggerer)*'. This is the GameObject that triggers the event and is automatically added to the receivers so you don't have to specify it.

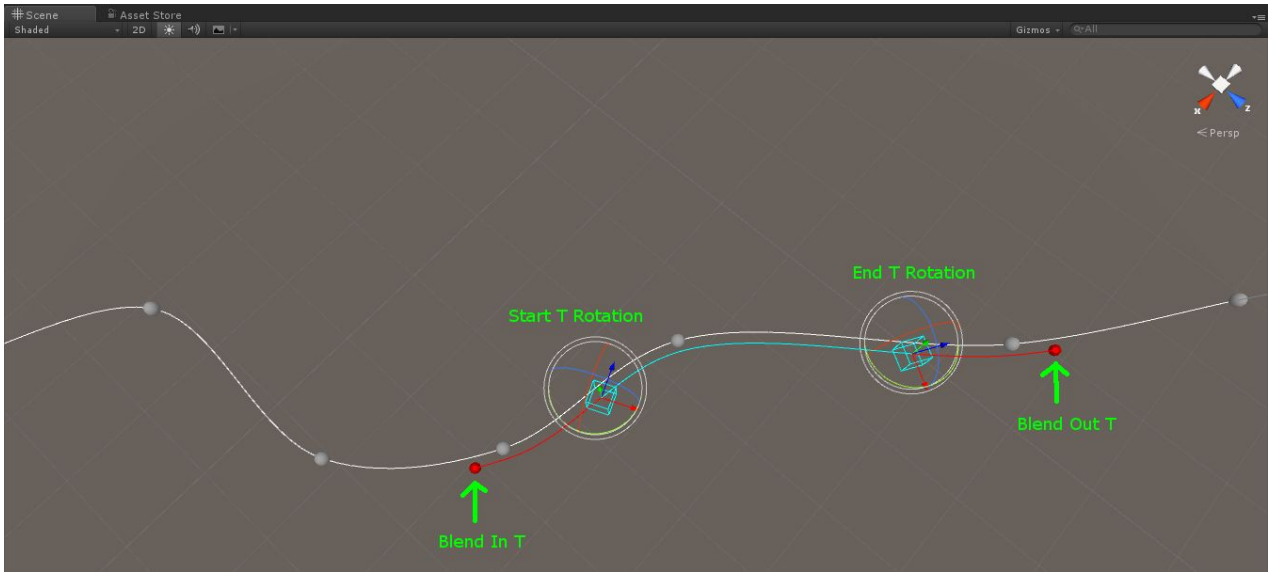
Defining Custom Rotations

Rotation Nodes can be used to define a custom rotation along a specified section of a spline. A rotation node is comprised of 4 “sub nodes” (or T values) as follows:

- *Blend In*: T value at which to start blending to the nodes “Start” rotation.
- *Start:* T value where the starting rotation is set, and blending to the ‘End’ rotation begins.
- *End*: T value where end rotation is set.
- *Blend Out*: T value at which the rotation will be completely blended back to the spline rotation.



To set up a custom rotation, add a rotation node on the spline using the spline inspector. Select the newly created rotation node in the hierarchy and adjust the 4 sub nodes to control where the rotation and blending will occur. Then use the transform tools in the scene editor to rotate the *Start* and *End* sub nodes to your desired rotations.

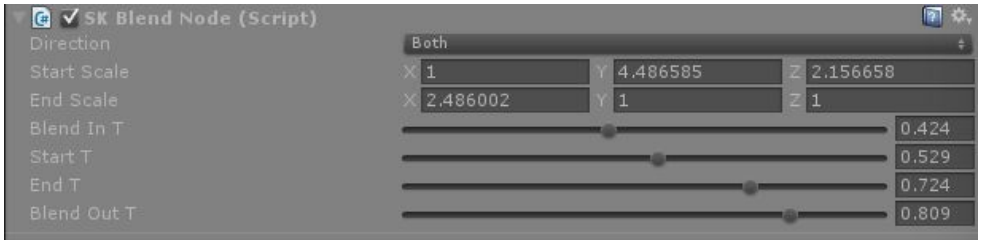


When a GameObject animating along a spline encounters a *‘Blend In’* sub node, it will start blending from it’s current rotation to the *‘Start’* sub node rotation. From here, it will blend from the *‘Start’* sub node rotation to the *‘End’* sub node rotation. And finally, it will blend from the *‘End’* sub node rotation back to the spline rotation at the *‘Blend Out’* sub node. Of course if you don’t want a any blend in or out, just set the corresponding T values equal to the Start/End T values. Additionally you can set the Start and End rotations to the same value to keep a constant rotation between them.

Defining Custom Scaling

Scale Nodes can be used to define a custom scaling along a specified section of a spline, much the same as Rotation nodes do for rotation. As such, a Scale node has the same 4 “sub nodes” (or T values) as follows:

- *Blend In*: T value at which to start blending to the nodes “Start” scale.
- *Start:* T value where the starting scale is set, and blending to the ‘End’ scale begins.
- *End*: T value where end scale is set.
- *Blend Out*: T value at which the scale will be completely blended back to the spline scale.



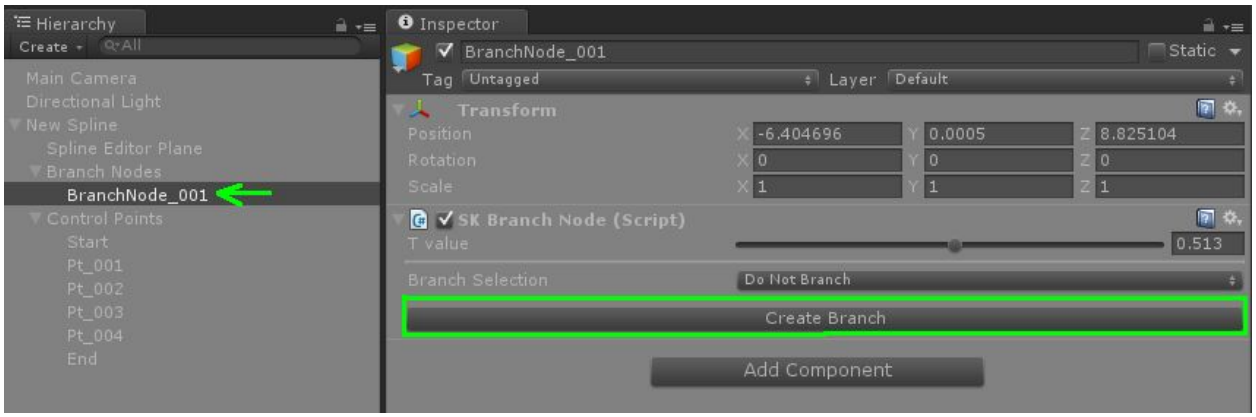
To set up a custom scale, add a scale node on the spline using the spline inspector. Select the newly created scale node in the hierarchy and adjust the 4 sub nodes to control where the scaling and blending will occur. Then use the transform tools in the scene editor to scale the *Start* and *End* sub nodes to your desired scale.

When a GameObject animating along a spline encounters a *‘Blend In’* sub node, it will start blending from it’s current scale to the *‘Start’* sub node scale. From here, it will blend from the *‘Start’* sub node scale to the *‘End’* sub node scale. And finally, it will blend from the *‘End’* sub node scale back to the spline scale at the *‘Blend Out’* sub node. Of course if you don’t want a any blend in or out, just set the corresponding T values equal to the Start/End T values. Additionally, you can set the Start and End scale to the same value to keep a constant scale between them.

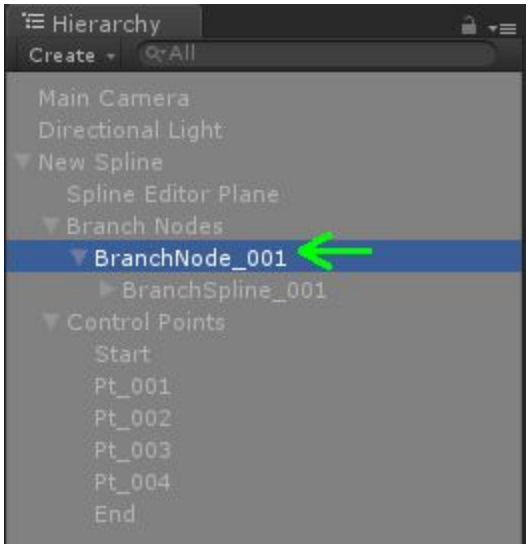
Creating Branches

Branch nodes can be used to create branches off of a spline and Anchor nodes can be used to join the branch back to its parent. To create a branch, first add a Branch node via the spline inspector.

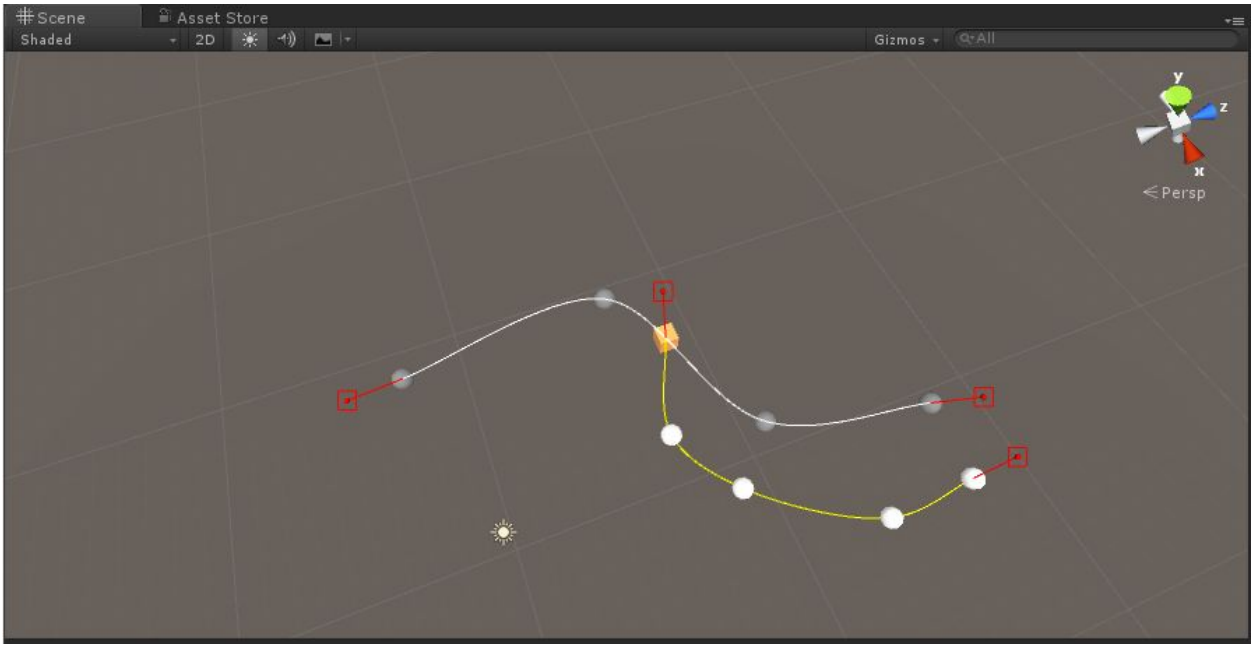
Then select the newly created Branch node and in its inspector click ‘Create Branch’.



This will create a new spline branch parented to the spline. You can now add points to the new branch spline just as you would a new spline, select the new branch spline



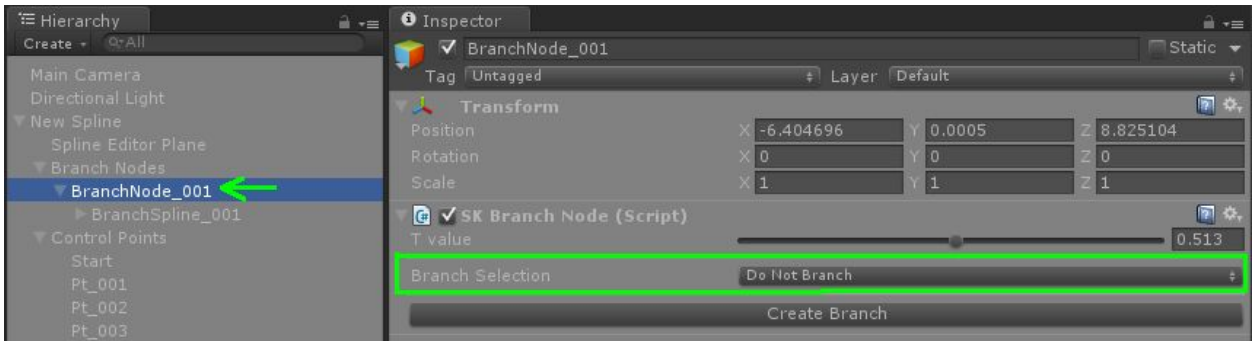
Then use Shift+LMB to add points in the scene view



Note: I set the branch color to yellow just to highlight it. This can be done via the color box in the spline inspector (color box to the right of 'Keep Spline Render Active').

To determine which path GameObjects travelling along the spline take, set the *'Branch Selection'* variable in the branch node inspector. The choices are

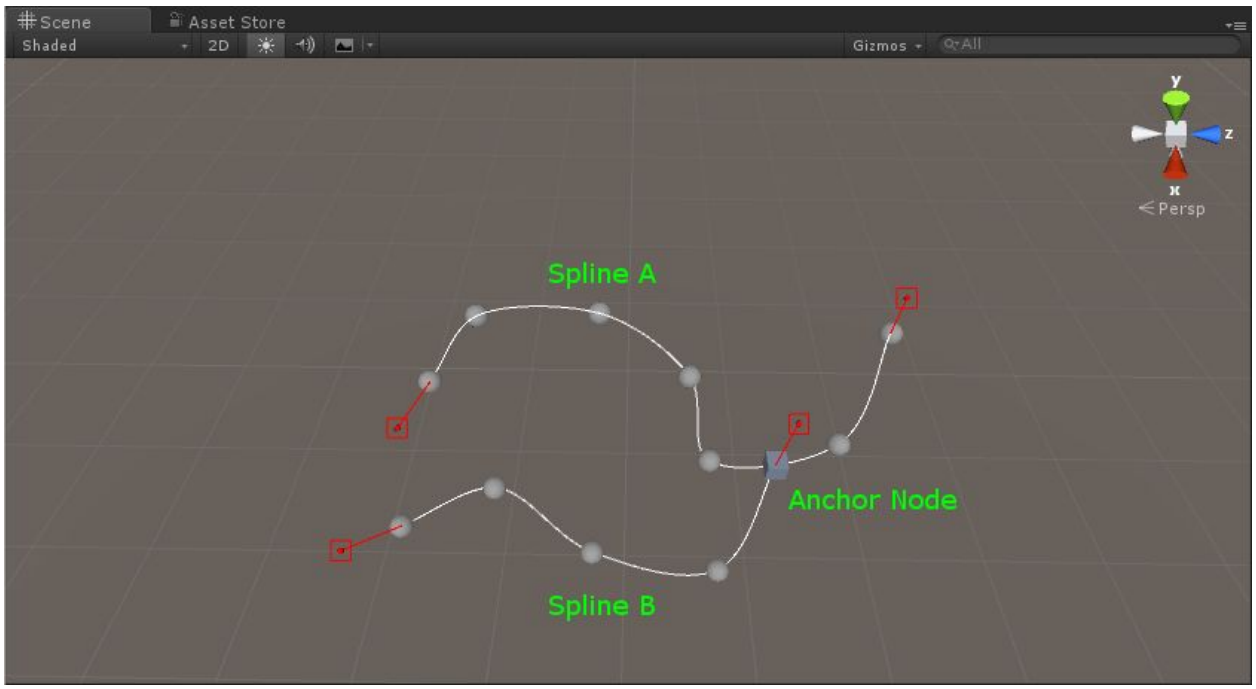
Do Not Branch	Don't branch, always stay on the main spline
Specify Branch	Specify the name of the path (main spline or branch spline) to take.
Random	Randomly select a path to take.
Random No Repeat	Randomly selects a path to take, guaranteeing that it will not take the same path twice in a row.
Execute Function	Execute your own logic to determine which path to take. Your function should take an *SKCmd* as a parameter and set *SKCmd.ReturnString* to the name of the path (main spline or branch spline) to take.

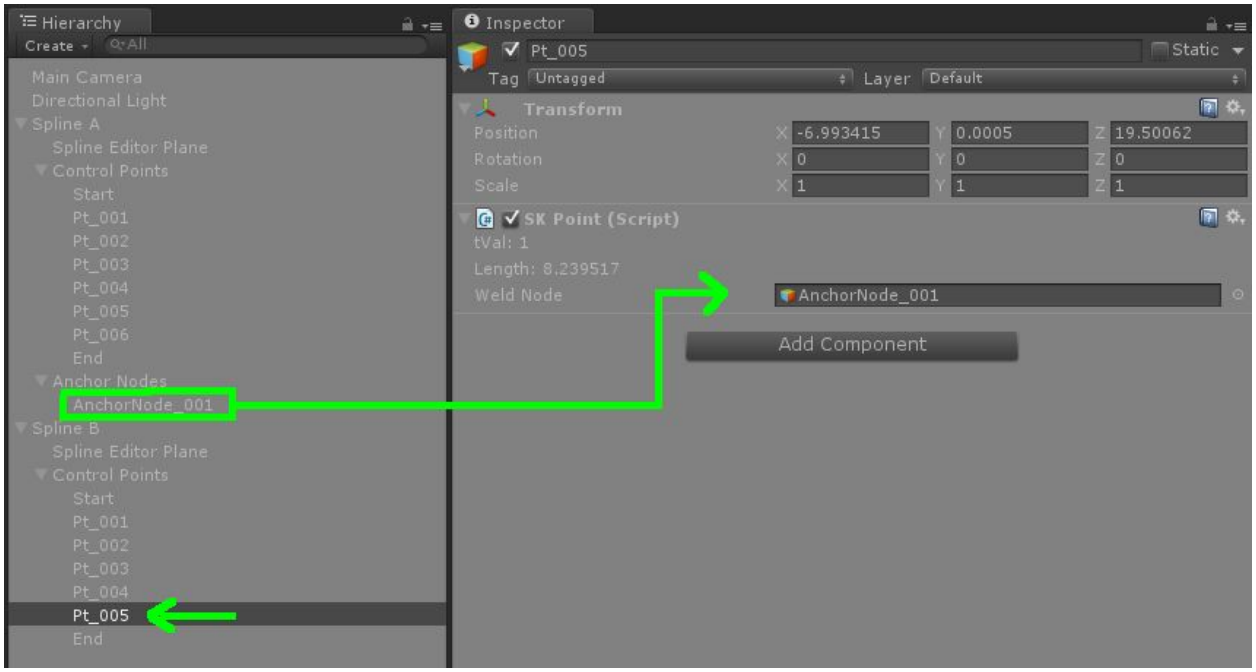


The branch can be reconnected to to the spline (or any other spline) by adding an anchor node to the other spline and welding the last point of the branch to it (*See Joining a spline to another spline*).

Joining a spline to another spline

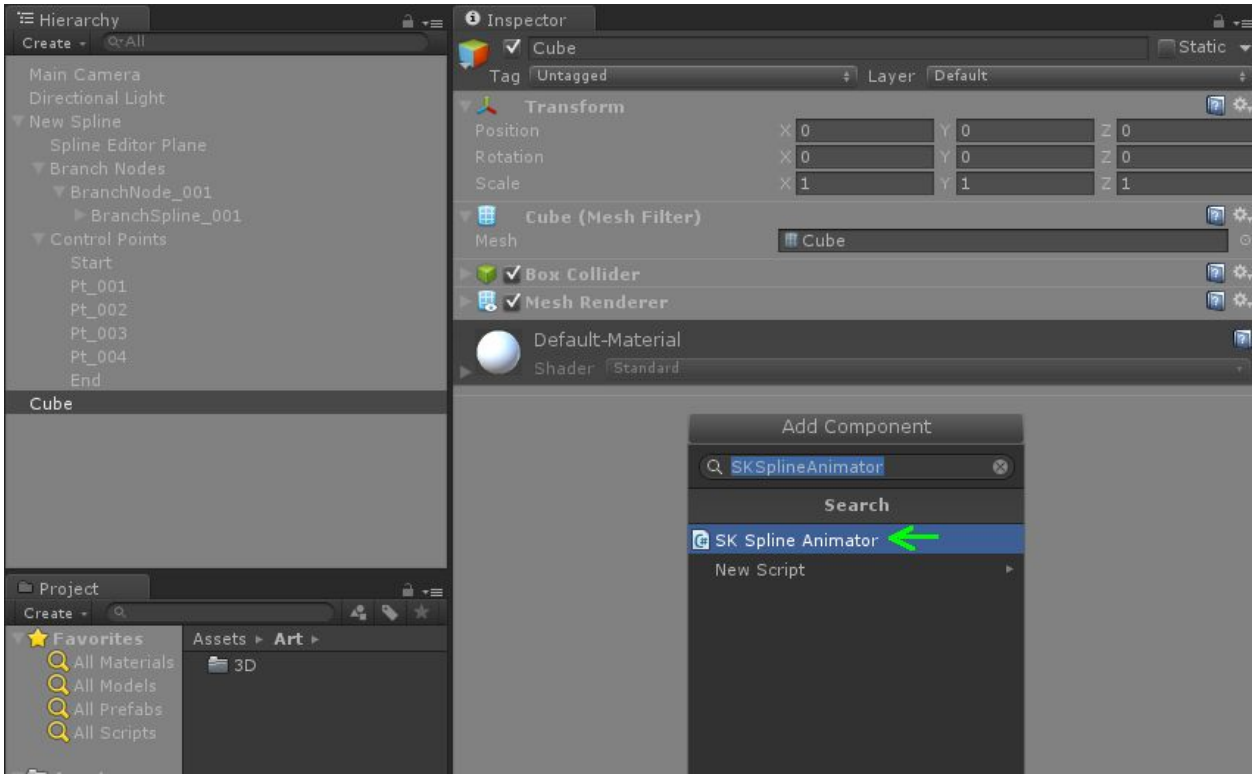
To join spline B to another spline A, first place an anchor on spline A where you'd like to spline B to join. Then select the last point of spline B (the point before the 'End' point in the hierarchy) and drag the anchor node from spline A into the 'Weld Node' of the last point of Spline B.



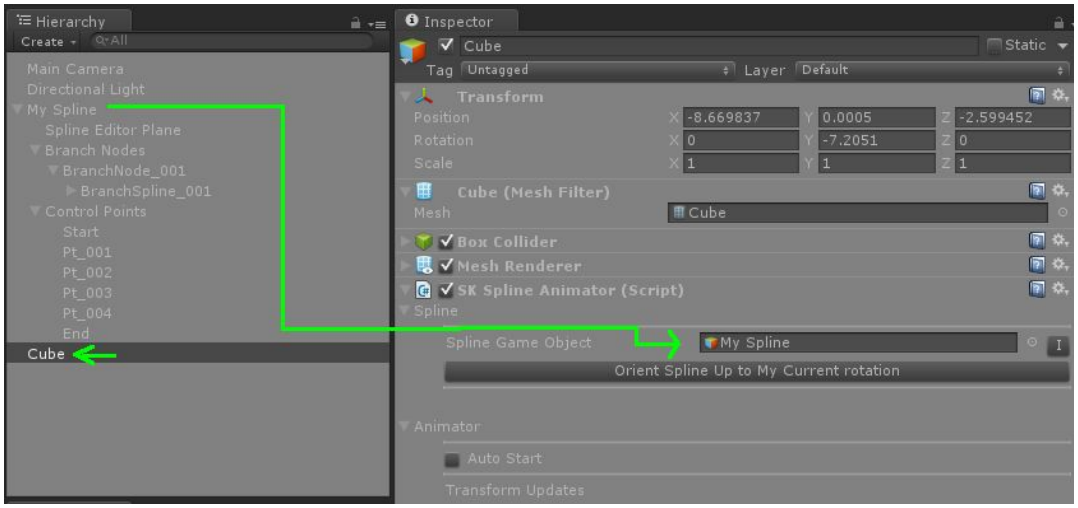


Animating a GameObject on a Spline

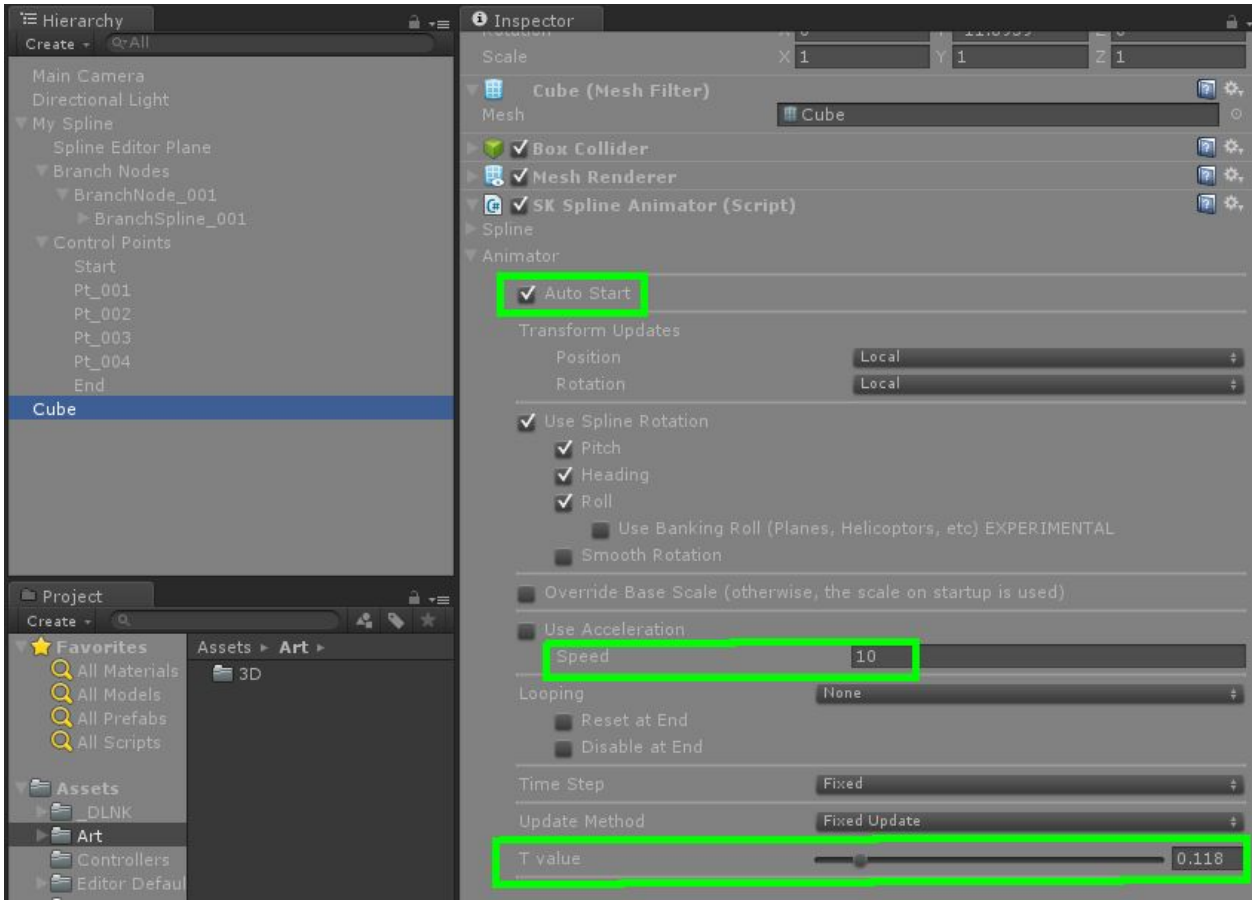
To animate a GameObject on a spline, you first need to add the SKSplineAnimator component to it. With your GameObject selected, click the Add Component button in the inspector and enter *SKSplineAnimator*.



To attach it to a spline, select your GameObject and drag the spline instance you which to attach it to into the *'Spline Game Object'* field in the inspector.



To test that it's working, select your GameObject and drag the '*T value'* slider under the SKSplineAnimator component and your object should move along the spline. You can also test it in play mode by checking *'Auto Start'*, giving it a positive speed, and hitting play.



SKSplineAnimator Component

The SKSplineAnimator component is where you set up how your GameObject will animate on the spline; such as, speed, rotation, and looping.

Spline

Spline Game Object	Reference to the spline you want to attach to. (The 'I' button will bring up a floating window with the spline inspector for easy reference).
Orient Spline Up to My Current Rotation	This is a convenience method that will set the spline up vector to match the current up vector of your GameObject. This is useful when you want the spline to keep your GameObject's up as it is.

Animator

Auto Start	If checked, the GameObject will start animating on on startup.
Use Adaptive Sampling	With normal spline evaluation it is normal to lose speed around tight turns. Use this if you need to keep speed consistent around tight turns. This is more performance intensive.
Sub Sample Scalar	This is how much to subdivide the step. So if you need to move 100 units in one frame, a scalar of 0.01 would keep evaluating the spline using a distance of 1 until 100 units was reached.
Transform Updates - Position	Whether the SKSplineAnimator should update the GameObject's global or local position.
Transform Updates - Rotation	Whether the SKSplineAnimator should update the GameObject's global or local rotation. (Has no effect if 'Use Spline Rotation' is unchecked).

Use Spline Rotation	Whether or not the spline should adjust the rotation of the GameObject.
Pitch	If checked, the pitch will be adjusted by the SKSplineAnimator.
Heading	If checked, the heading will be adjusted by the SKSplineAnimator.
Roll	If checked, the roll will be adjusted by the SKSplineAnimator.
Use Banking Roll	Experimental. If checked, the SKSplineAnimator will attempt to determine the bank of the spline and apply it to the roll (useful for things like planes, helicopters, etc).
Smooth Rotation	If checked the SKSplineAnimator will smooth the rotation from it's current to the spline rotation.
Smoother	How much to smooth each frame.
Override Base Scale	By default the SKSplineAnimator uses the scale of the GameObject on startup to determine the base scale (the scale to return to when using Scale Nodes). Use this if you want to set your own base scale.
Base Scale	The base scale used when scaling the GameObject on the spline.
Speed	The speed (or initial speed if using acceleration) of the GameObject.
Acceleration	The rate of acceleration
Min/Max Speed	The max (or min speed if acceleration is negative) for the GameObject; meaning the GameObject's speed will never exceed this value.
Looping	None - The GameObject will not loop. Circular - The GameObject will continue from the beginning after reaching the end of the spline. Reverse - The GameObject will reverse back along the spline when reaching the end or beginning of the spline.

Reset at End	If checked the GameObject will be positioned at the beginning of the spline after reaching the end (Only available when looping is set to ‘None’).
Disable at End	If checked the GameObject will be deactivated when reaching the end of the spline (Only available when looping is set to ‘None’).
Time Step	The time step to use when updating the GameObject. Possible values are: ● Variable - Uses the game’s variable time step. ● Fixed - Uses the fixed interval that’s used for updating physics. ● Smoothed - Uses the unity smoothed delta time step. ● Unscaled - Uses the unscaled time it took to complete the last frame.
Update Method	When the update of the GameObject should occur: ● Update - Updates the GameObject in the same frame as other GameObjects. ● Fixed Update - Updates in the same frame as the physics are updated. ● Late Update - Updates after all of the GameObjects and their physics updates have been updated.
T Value	The starting normalized time along the spline. This can also be used to “scrub” the GameObject along the spline in the Scene View.

Using the Lob Generator

The SKLobGenerator can be used to dynamically create lob paths that start from a specified locator and terminate at a specified target (or a randomly picked target). For example, let's say you have a character that needs to lob a grenade and have it land in a manhole. With the SKLobGenerator you can tell it to create a path from the character's hand to the center of the manhole and lob the grenade along it. Additionally, you can chain lob paths to create "bounces"; for example, your grenade could make a couple random bounces on the ground before making it into the manhole.

Inspector

Destroy Paths on Complete	If checked, the generated splines will be destroyed when the animated GameObject reaches the end of the spline.
Destroy Animator on Complete	If checked, the animated GameObject will be destroyed when it reached the end of the spline (or spline chain).
Initial Velocity	The initial velocity of the lob
Simulate Gravity	If 0 is specified, then the velocity will remain constant. If a non zero value is specified, it will be applied as an acceleration/deceleration to the lob (when the lob is moving upward, in reference to the lob path up vector, it will decelerate the lob, otherwise, it will accelerates).
Delay	The delay before starting the animator along the spline. If 0, it starts immediately.
Lob Paths	A list of lob paths. Each path specifies a start locator and 1 (or more) end locators. If more than one end locator is specified, the lob generator will randomly pick one to use. You can specify more than one lob path if you want to chain more than one together. *(See Lob Path below for more details).*

Lob Path

Start Locator	A transform or GameObject reference to where the lob should begin
End Locators	1 or more transforms where the lob can end. If more than 1 is specified, then the lob generator will choose 1 at random.
Normalized Mid Point Distance	The normalized distance (0.0-1.0) along the generated spline where the lob should peak. In other words, a value of 0.5 will have the object peak in the middle of the spline.
Min Mid Point Height	Minimum height of the peak.
Max Mid Point Height	Maximum height of the peak. If you always want the lob to peak at the same height, set the min and max to the same value; otherwise, the peak will be a random height between the min and max.
Up	The vector that is up for you (usually 0,1,0). The peak will be offset in this direction.
Hit PFX Prefab	If specified, this prefab will be instantiated each time the object reaches the end of a lob path (for example, it could be a particle effect, that plays on each bounce, but it could also be anything you want).
Destroy Hit PFX after	This specifies the number of seconds that the instance of the hit pfx should live. If you specify 0, it won't be destroyed.

Extracting Animation Curves for Motion

Animation curves can be extracted and used to control the speed of your GameObject along the spline. This is useful, for example, if an animator animates the stop motion for a character and you want to match it, so the footsteps don't fall out of sync with the animation.

To use an animation curve, you first need to extract it into a container for use. To do this, open the *Spline Kit Pro Animation Curve Extractor* (*Spline Kit Pro->Animation Curve Extractor*), drag in the animation clip, tell it which animated property you want to use, and hit 'Extract'. This will create a scene object called *SKExtractedCurve* that you can use in a spline trigger command.

Properties

Animation Clip	The animation clip you want to extract from.
Property Path	This is the Unity animation path you see when you look in the 'Animation' Window. Most animations have a root node, so in the animation window you might see something like 'ROOT : Position', here 'ROOT' is the path. If the animation was imported and has a rig, the path is usually the bones you'd traverse to get to the property. Let's say you had a character with bones Root->Pelvis->LLeg and you wanted to use an animated property from the Left leg (LLeg). The Path would be Root/Pelvis/LLeg
Property Name	This is name of the animation property you want to extract. So for 'ROOT : Position', position is the property, but, there are probably 3 channels (x, y, and z) and you need to specify that as well. So if you expanded 'ROOT : Position' you might see Position.x, Position.y, Position.z. So if you wanted to extract the Z motion, you'd use Position.z
Playback speed	The playback speed of the animation you're trying to match. In other words, if the animation you're trying to match plays at a different speed, then set it here.

List Properties	Lists all the property paths and property names for the given animation clip.
Extract	Extracts the requested curve.

Using Animation Curves for Motion

Once you've extracted an animation curve you'd like to use, you can use it to control the speed of a GameObject on a spline. To do this, set up a trigger where you'd like the curve to take effect. Then add '*Apply Motion Curve'* command and drag in your extracted curve.

Tips and Tricks

- Orienting your Game Object correctly on a spline when its Z axis is not its forward. This is easy to correct, but also easy to mess up. Order of operations is important!

    1. Create an empty game object and rotate it such that it would correct the rotation of your object. For example, if we had a object facing backwards (such that it looks down it's -Z) we would need it to be rotated 180 degrees around the Y; so, we'd create an empty game object with 180 rotation on Y.

    2. Then, ***after*** you have rotated the empty game object, parent your object to the empty one, this way it keeps it's world rotation. If we parent it before, it will be rotated when we rotate the parent and it won't work.

    3. Add the SKSplineAnimator component to the new empty Game Object parent, not to your original Game Object. The object will now be oriented correctly.

- Your Game Object's center is not at its feet. Use the same technique as above, just with translation.

Tutorials

We've created a series of videos to help you get the most out of Spline Kit Pro. You can view these at http://rmotion.wixsite.com/home/videos (or on our you tube channel : SplineKit).

Support

http://rmotion.wixsite.com/home/contact