Computer Graphics
Section 2 (25-02-2018)

Unity Animation

Objectives

How to animate and work with imported animations.

Animation Overflow

Unity's animation system is based on the concept of **Animation Clips**, which contain information about how certain objects should change their position, rotation, or other properties over time.
Each clip can be thought of as a single linear recording.

Animation Clips are then organized into a structured flowchart-like system called an **Animator Controller**. The Animator Controller acts as a "State Machine" which keeps track of which clip should currently be playing, and when the animations should change or blend together.

Unity's Animation system also has numerous special features for handling humanoid characters which give you the ability to retarget humanoid animation from any to your own character model, as well as adjusting muscle definitions. These special features are enabled by Unity's **Avatar system**, where humanoid characters are mapped to a common internal format.

Each of these pieces - the **Animation Clips**, the **Animator Controller**, and the **Avatar**, are brought together on a GameObject via the **Animator Component**.
This component has a reference to an Animator Controller, and (if required) the Avatar for this model. The Animator Controller, in turn, contains the references to the Animation Clips it uses.

Animation Clips and Animation View

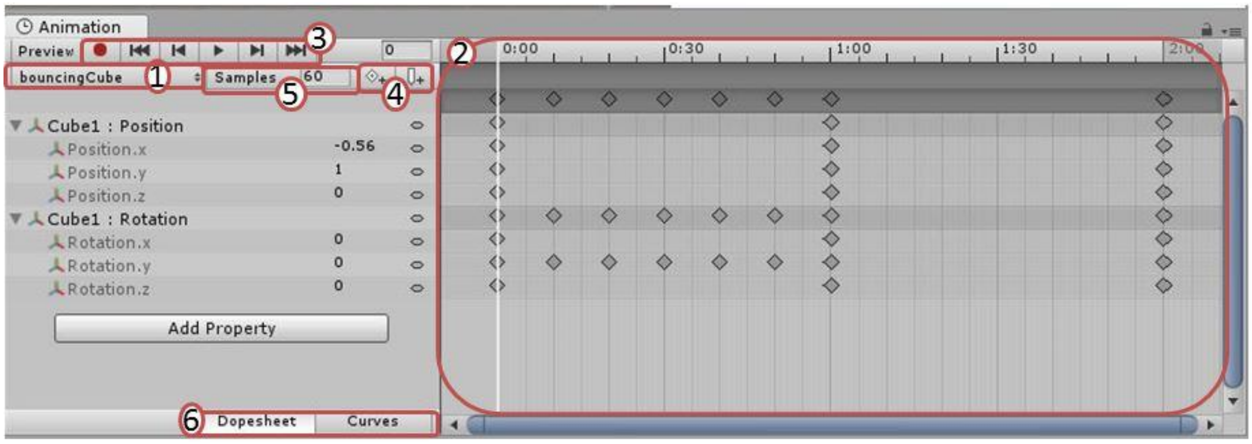
The **Animation Window** in Unity allows you to create and modify **Animation Clips** directly inside Unity.
In addition to animating movement, the editor also allows you to

- Animate variables of materials and components.
- Augment your Animation Clips with Animation Events, functions that are called at specified points along the timeline.

What's the difference between the Animation window and the Timeline window?

The Timeline Window	The Animation Window
Allows you to create cinematic content, game-play sequence, audio sequence and complex particle effects.	Allows you to create individual animation clips as well as viewing imported animation clips
You can animate many different GameObjects within the same sequence.	Animation clips store animation for a single GameObject or a single hierarchy of GameObjects.
In the Timeline window you can have multiple types of **track**, and each track can contain multiple **clips** that can be moved, trimmed and blended between.	The Animation Window can only show one animation clip at a time.
It is useful for creating complex animated sequence that require many different GameObjects to be choreograpged together.	It is useful for animating discrete items in your games such as sliding door, or spinning coin.

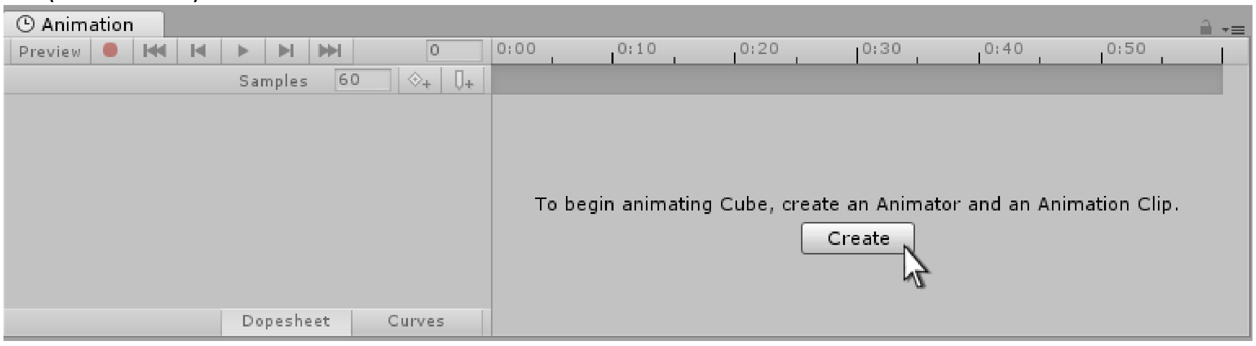
There are many different buttons and different options visible when you first open the Animation View.



1. Animation Dropdown
   We use it to select out different animation to edit.
2. The Timeline
   is where you will be able to establish the timing of your animation.
   The Timeline is measured in frames and seconds.
   The numbers in the time line are shown as seconds and frames, so 1:30 means 1 second and 30 frames.
3. The **Record** and **Play** buttons
   When the **Record** button is pressed, changes made to the chosen object in the scene view will be added automatically to the animation of the currently selected frame.
   The **Play** button will allow you to preview your animation.
   The **Left** and **Right** arrows
   Allows you to navigate through the keyframes of the animation.
4. Other buttons which allows you to add Keyframe and events to the currently selected frame.
5. The Sample property
   Determines how many frame make up one second of animation.
   Reducing this number makes our animation slower.
6. At the bottom there are two buttons that allow you to transition between **Dopesheet** mode and **Curves** view mode.

Exercise 1

- Create a new Project in Unity in the 3D mode.
- Add a Cube to the Scene
  - Change its name to *ground*.
  - Change its Transform component, position property, ***x*** value and ***z*** value to 10;
- Add another cube to the scene
  - Change its name to *Cube1*.
  - Change its Transform component, position property, ***y*** value to 1 and change **z** value to -3.
- Open the Animation View (if it is not already opened)
  - To open the Animation go to **Window** > **Animation**.



To begin animating Cube, create an Animator and an Animation Clip.

Figure 1 Animation Window (View)

- Click the **Create** button to create a new animation clip.
  When you create a new animation you will be prompted for a **name** and a **location** to save the animation file.
  Once you have saved this new empty Animation Clip, a number of things happen automatically:
    - A new Animator Controller asset will be created
    - The new clip being created will be added into the Animator Controller as the default state
    - An Animator Component will be added to the GameObject being animated
    - The Animator Component will have the new Animator Controller assigned to it.
      The result of this automatic sequence is that all the required elements of the animation system are set up for you, and you can now begin animating the objects.

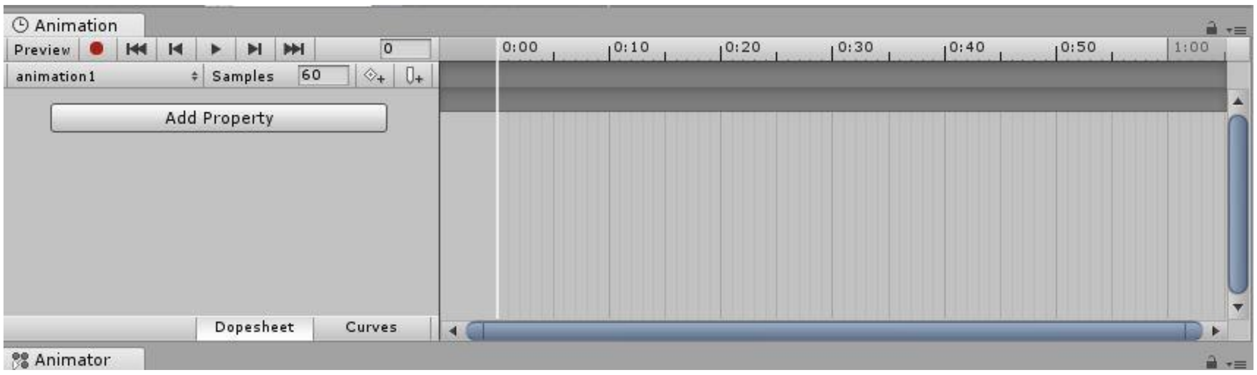


Figure 2 Animation Window After creating and saving an Animation Clip

- Click the **Animation Record button** to begin recording keyframes for the selected GameObject.
  This enters **Animation Record Mode**, where changes to the GameObject are recorded into the **Animation Clip**.
  Once in **Record mode** you can create keyframes by setting the white Playback head to the desired time in the Animation time line, and then modify your GameObject to the state you want it to be at that point in time.
    - Select frame number 60,then change the Transformation Component, position property, ***y*** value to 4.
      Notice what happened.
        - Two keyframs has been created one in frame 0 and the second one in freame 60.
        - A new **Property** has been created (at the left of the Animation Window)
          Transform properties are special in that the **.x**, **.y**, and **.z** properties are linked, so curves for all three are added at the same time.

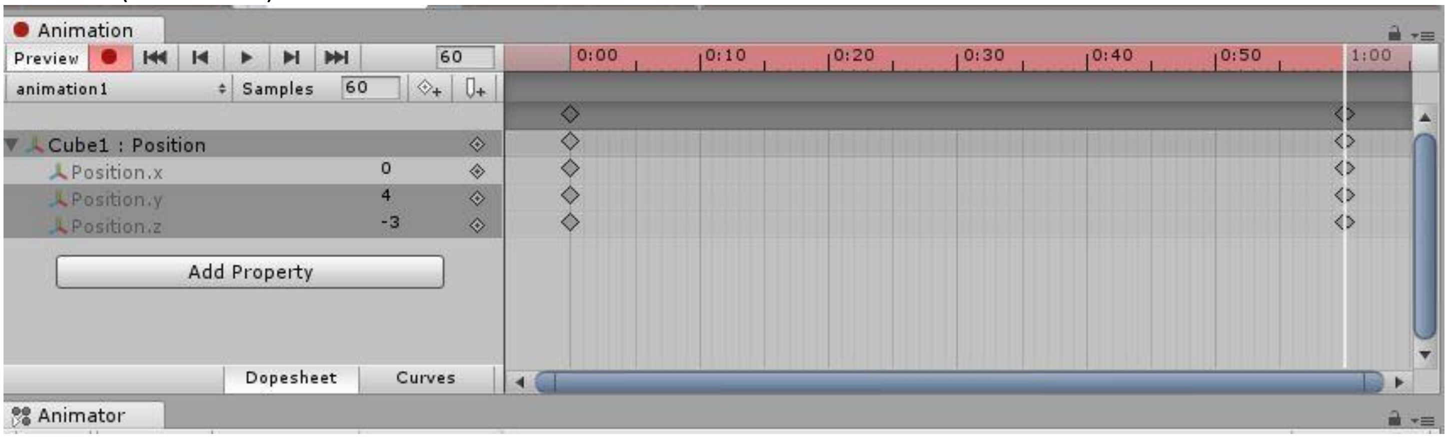


Figure 3 Animation Window while in Recording Mode

- Stop the **Animation Record Mode** by clicking again in its button .
- Press the play button to see what you created.

* Now we want to add a keyframe manually to change the position of the *cube1* bake to its initial value.
    - Select the frame number 120
    - Click the Add Keyframe button
    - Change the Position.y value back to 1.
    - Click the play button to see that happened.
* To fine tune our animation we use the **Curves** mode.
    - Click on the Curves button on the bottom left of the animation window.
      Here we can see the various curves of our animation by selecting the corresponding component on the left.
      To modify a value on a curve, click and drag a keyframe to the desired value.
      To create a new keyframe click the create keyframe button, or double click on a curve.
    - We can see that the *Cube1* height is **interpolated** in-between the keyframes.
      You notice that the default interpolation is not quite linear.

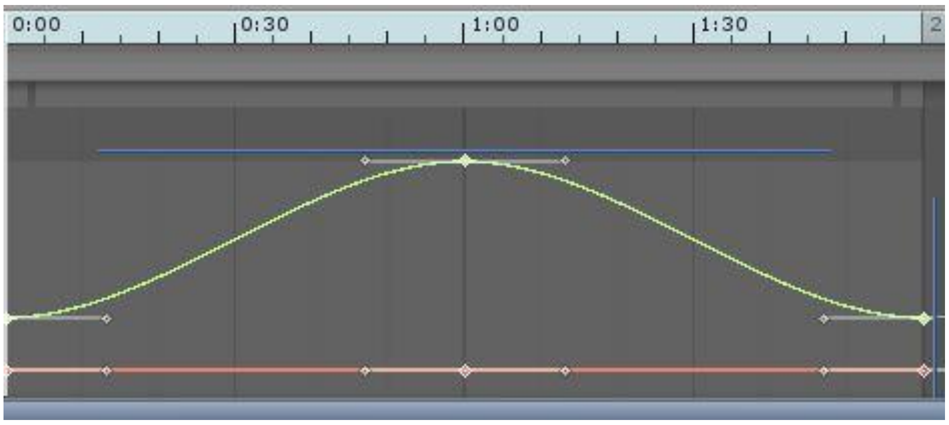


Figure 4 The Default interpolation

The default setting for key is **Clamped Auto**.

Editing Tangents
A key has two tangents - one on the left for the incoming slope and one on the right for the outgoing slope. The tangents control the shape of the curve between the keys. You can select from a number of different tangent types to control how your curve leaves one key and arrives at the next key.
Tip: Right-click a key to select the tangent type for that key.
- Clamped Auto: This is the default tangent mode. The tangents are automatically set to make the curve pass smoothly through the key. When editing the key's position or time, the tangent adjusts to prevent the curve from "overshooting" the target value. If you manually adjust the tangents of a key in Clamped Auto mode, it is switched to Free Smooth mode.

- Select all keyframes by pressing **ctrl+A** and then right click on part of the time line, then choose Both Tangents > choose linear.

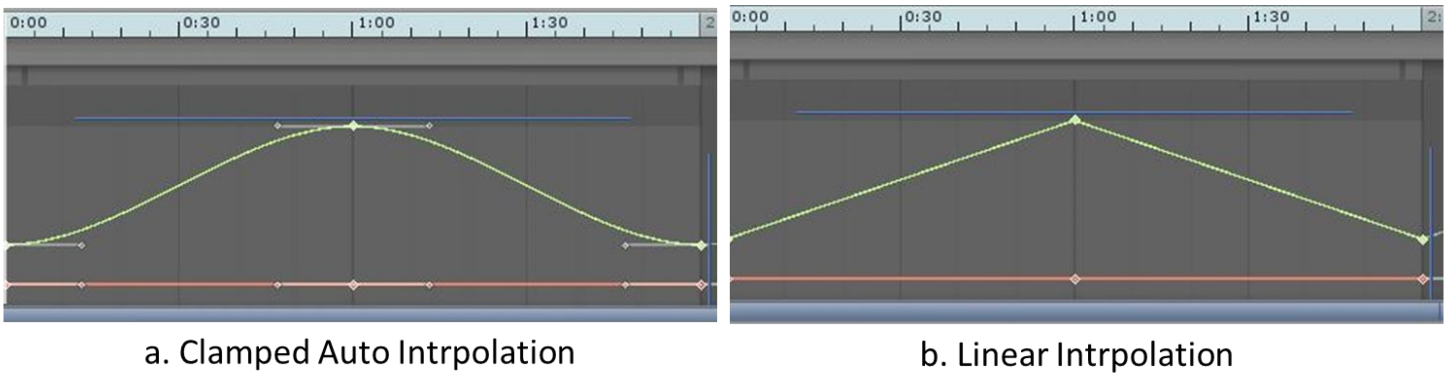


Figure 5

- Play the animation and try to observe the difference.

Exercise 2
We know that there are 12 Principle of Animation. One of them is **Timing**.
When animating to mimic physical movement, how much time each action takes is very important.
Picture an object accelerating , if it were move from left to right, we would draw more frames toward the left side when it is moving slower and fewer frames on the right when it is moving quickly.

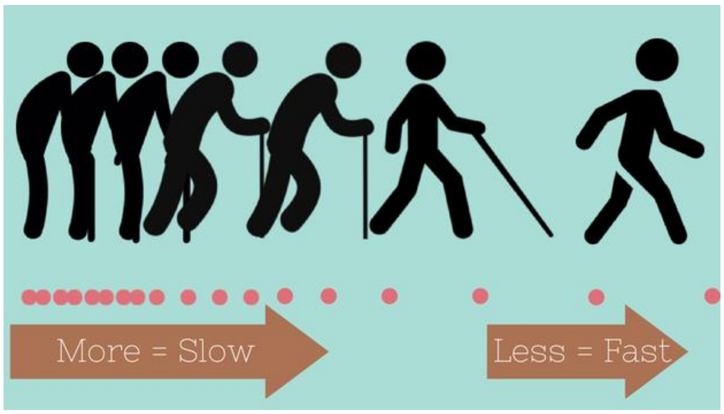


Figure 6

In the case of our jumping cube, the critical actions are jumping time, hang time and impact time. Assuming that there is no loss of energy.
When the cube start jumping it will accelerate, the point at which it is moving fast would be the moment after start jumping (few frames). Its speed will drop until it reaches the hang position where it will stop momentarily (more frames) . After the momentarily suspend in air it will accelerate down due to the force of gravity, the point at which it will be moving fastest would be the instant before it strikes the ground.

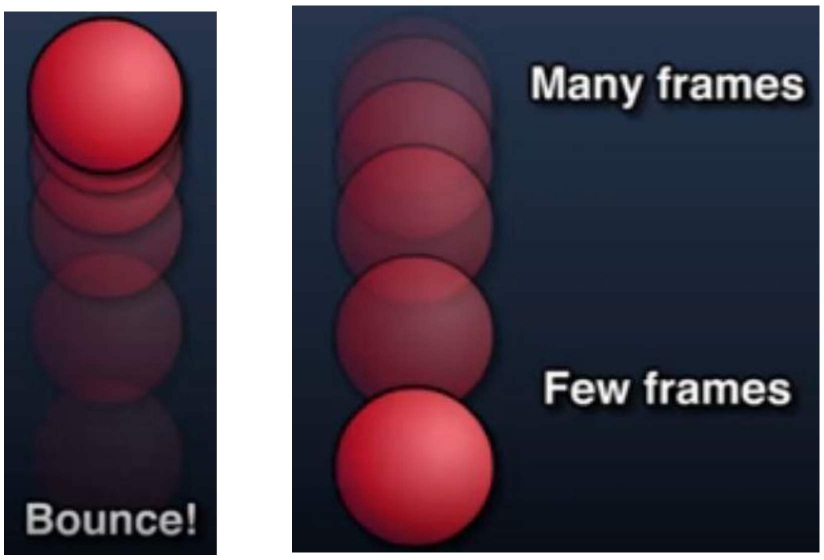


Figure 7 the second image on the right is a falling ball

Let's fix the time of our animation to better reflect how physics would act on it.

- We want the Cube to start jumping fast and then its speed will decrease till it stop in the hang position. To do that we will change frame number 60, so we will set this keyframe to **flat**.

Editing Tangents
- **Flat**: The tangents are set to be horizontal (this is a special case of Free Smooth).

- We will switch the tangent of the **y**-value at the jumping point to **broken**-**free**.

Editing Tangents

-- Sometimes you might not want the curve to be smooth when passing through a key. To create sharp changes in the curve, choose one of the **Broken** tangent modes. When using broken tangents, the left and right tangent can be set individually. Each of the left and right tangents can be set to one of the following:

- Broken-Free: Drag the tangent handle to freely set the tangents.
- Broken-Linear: The tangent points towards the neighboring key. TO makea linear curve segment, set the tangents at both ends to be **Linear.**
- Broken-Constant: The curve retains a constant value between two keys. The value of the left key determines the value of the curve segment.

- Now we can move these handle to manipulate the curve by *bringing in* these handles or allocating fewer frames to the time around impact and more frames to the hang time.
- We can fix the point of impact action in the same way.
  The final curve of the y-value should look like the figure below.

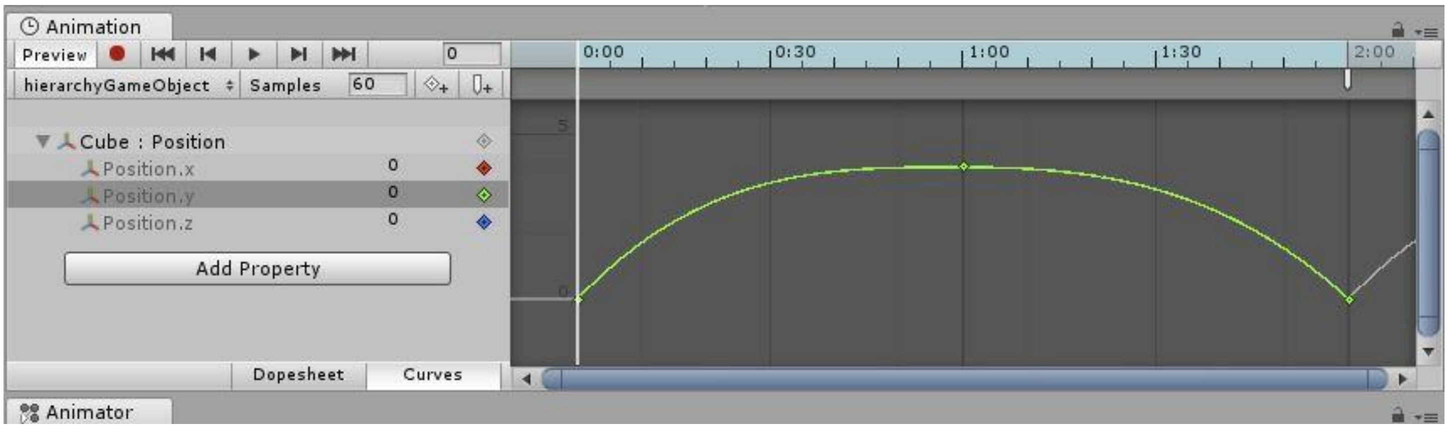


Figure 8

Exercise 3

Now let’s try to manipulate another property but this time we want to use the **Curves** Mode.
Let’s try to manipulate the **Rotation** property. We want to make to cube to rotate form 0 (when start jumping) to 180 (when it reaches the hang point) and then continue rotation to 360 while falling

- Select the Cube we want to add the rotation property to it.
- Click the **Add Property** button, choose Transform from the opened menu then choose Rotation and click the Plus sign.

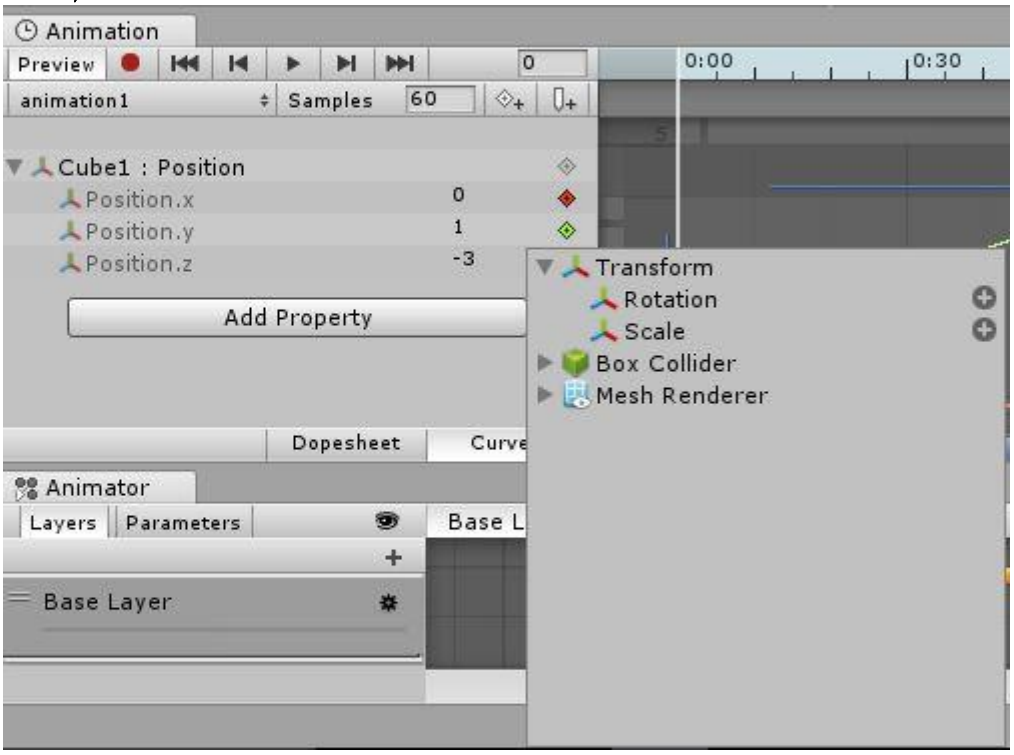


Figure 9

- Expand the rotation property; we will manipulate the rotation over **y-axis**.
- Choose **Rotation.y** from the property list.
- Select frame number 60 and change the Rotation.y value to 180.
- Select frame number 120 and change the Rotation.y value to 360.
- Play the animation and observe the changes.