MODUL 4 – 2D Animation

A. OBJECTIVES

- Student may be able to create an animation in 2D object.
- Student may be able to animate in certain GameObject component.
- Student may be able to understand in state function in game development.
- Student may be able to implement a trigger scheme in game development.
- Student may be able to create an animation from sprite sheet sequence.

B. GUIDANCE

1. Start your activity with a praying.
2. Read and study your material carefully.
3. Do your assignment with full passionate.
4. You may ask your lecture if something is not clear.

C. TIME ALLOCATION: 4 hours

D. THEORY

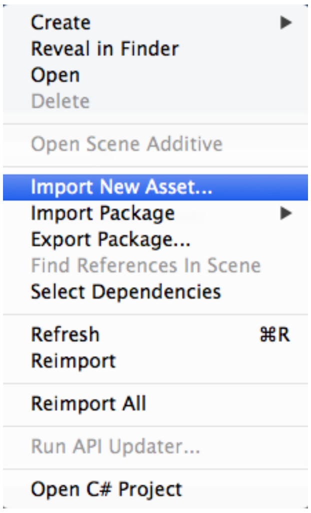
Animation is one of many components in game development. In this activity, we will learn about making animation which consists of two things, such:

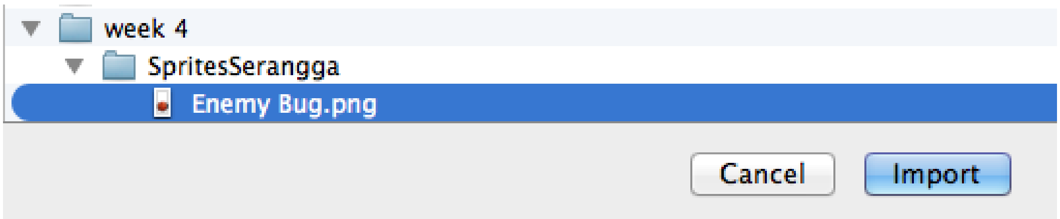
- State is a sequence of some representations to solve the problem. In game animation, state is used to set the story of animation.
- Trigger is used to make the game more exciting. In this module, trigger is also implemented in animation, for example to make a behaviour for an item is dropped by a Player action.

E. ACTIVITIES

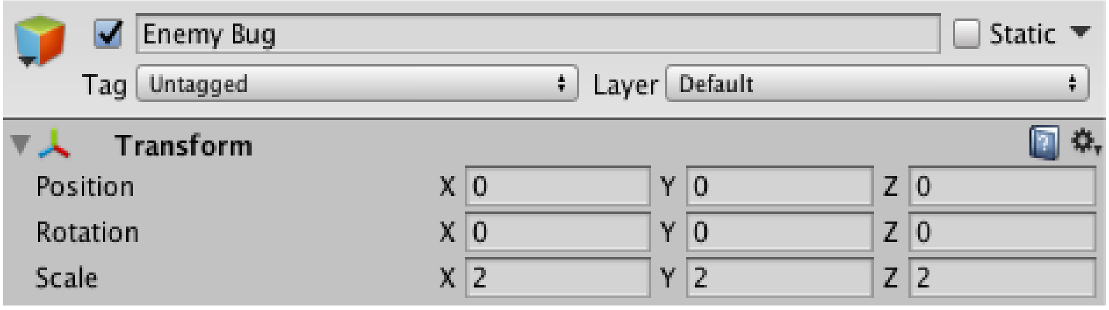
1. Flipping Sprite Horizontally

1. Create a new 2D Unity project.
2. Import EnemyBug.png to your project.





3. Drag Enemy Bug to scene. Set the position (x,y,z) of this GameObject to (0, 0, 0) then set the scale to (2, 2, 2).


☑ Enemy Bug □ Static

Tag Untagged Layer Default

Transform

Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	2	Y	2	Z	2

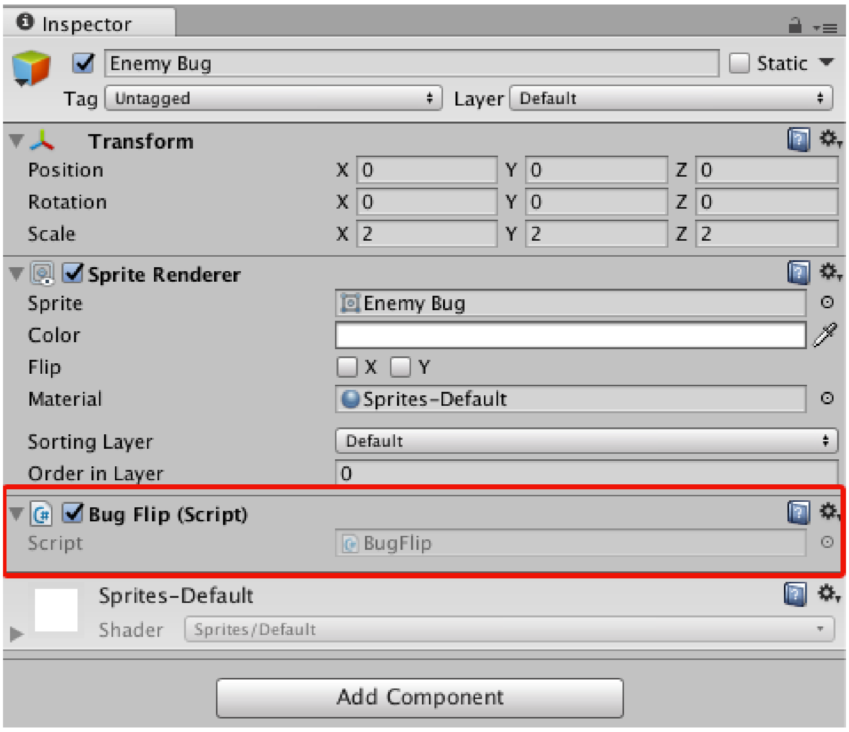
4. Create a new C# script named BugFlip with this following code, then add this code to be component of GameObject Enemy Bug.

```
using System.Collections;
using UnityEngine;

public class BugFlip : MonoBehaviour {
    private bool facingRight = true;

    // Update is called once per frame
    void Update() {
        if (Input.GetKeyDown(KeyCode.LeftArrow) && facingRight)
            Flip ();
        if (Input.GetKeyDown(KeyCode.RightArrow) && !facingRight)
            Flip();
    }

    void Flip (){
        // Switch the way the player is labelled as facing.
        facingRight = !facingRight;
        // Multiply the player's x local scale by -1.
        Vector3 theScale = transform.localScale;
        theScale.x *= -1;
        transform.localScale = theScale;
    }
}
```



☑ Enemy Bug □ Static

Tag Untagged Layer Default

Transform

Position	X 0	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 2	Y 2	Z 2

☑ **Sprite Renderer**

Sprite: Enemy Bug

Color:

Flip: □ X □ Y

Material: Sprites-Default

Sorting Layer: Default

Order in Layer: 0

☑ **Bug Flip (Script)**

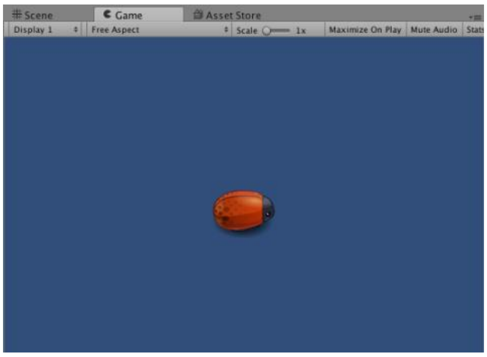
Script: BugFlip

Sprites-Default

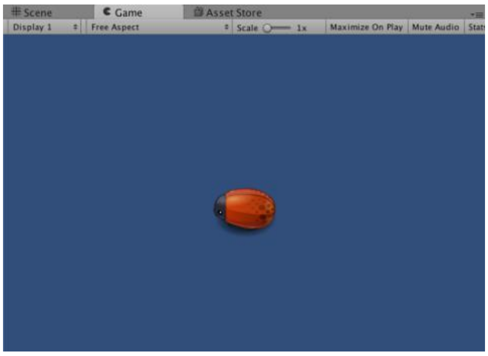
Shader: Sprites/Default

5. Run your project. Try to push your left or right arrow button on your keyboard, then you will see that the GameObject will change the direction based on which button is pressed.

Tekan button arah ke kanan



When right arrow button is pressed



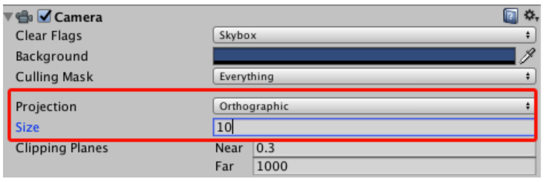
When left arrow button is pressed

6. Save your scene.

2. Animation for body parts when caharacter is moving

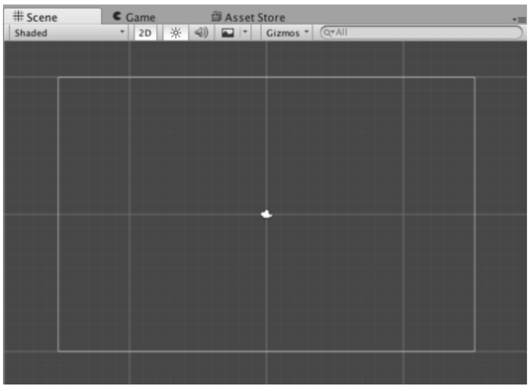
In this activity, we will learn about making an animation for boyd parts when a character is moving. It useful to make your character be more realistic. To do this, lets follow this some instructions:

1. Create a new 2D Unity project.
2. Import asset from PotatoMan2Dassets folder.
3. Set the size of **Main Camera** to 10.

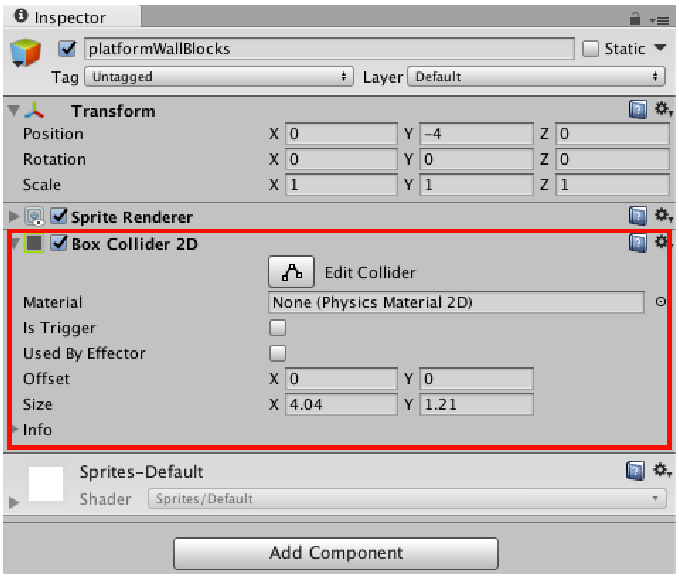


Camera
Clear Flags: Skybox
Background
Culling Mask: Everything
Projection: Orthographic
Size: 10
Clipping Planes: Near 0.3, Far 1000

Your scene will look bigger



4. Set 2D gravity by **Edit | Project Settings | Physics 2D**, then set Y value to -30.
5. Drag **character2D** to scene. Set position (x,y,z) of GameObject to (0, 3, 0).
6. Drag sprite platformWallBlocks from **Project | Sprites** to scene. Set position of GameObject to (0, -4, 0).
7. Add Box Collider 2D component to platformWallBlocks by **Add Component | Physics 2D | Box Collider 2D**.



Inspector

☑ platformWallBlocks □ Static

Tag Untagged Layer Default

Transform

Position	X	0	Y	-4	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

☑ Sprite Renderer

☑ Box Collider 2D

Edit Collider

Material	None (Physics Material 2D)			
Is Trigger	□			
Used By Effector	□			
Offset	X	0	Y	0
Size	X	4.04	Y	1.21

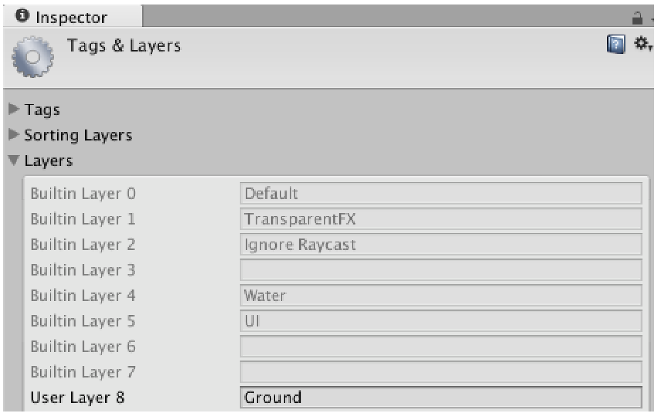
Info

Sprites-Default

Shader Sprites/Default

Add Component

8. Change the layer of platformWallBlocks by Add layer the fill the column with value: Ground.



Inspector

Tags & Layers

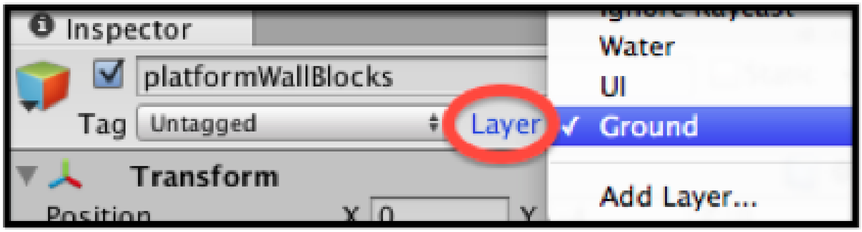
▶ Tags

▶ Sorting Layers

▼ Layers

Builtin Layer 0	Default
Builtin Layer 1	TransparentFX
Builtin Layer 2	Ignore Raycast
Builtin Layer 3	
Builtin Layer 4	Water
Builtin Layer 5	UI
Builtin Layer 6	
Builtin Layer 7	
User Layer 8	Ground

Back to platformWallBlocks in hierarchy, then select Ground layer.

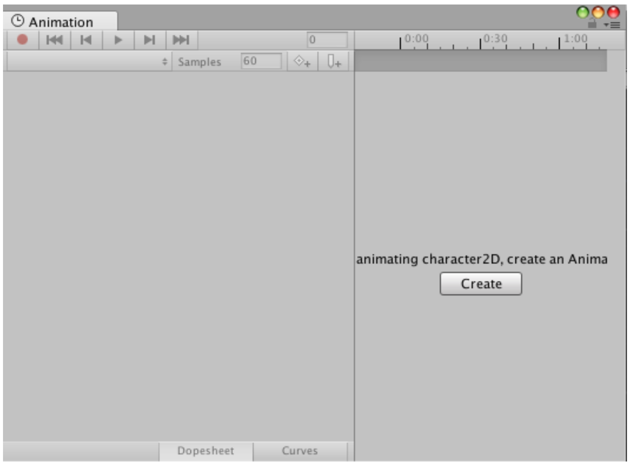


Inspector

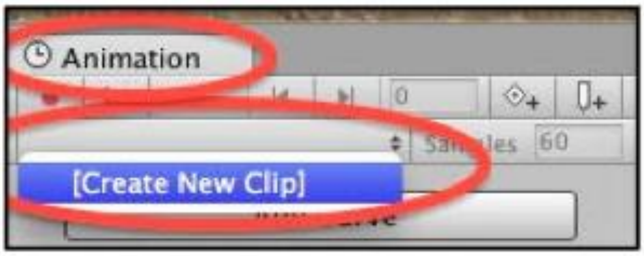
☑ platformWallBlocks

Tag Untagged Layer ✓ Ground

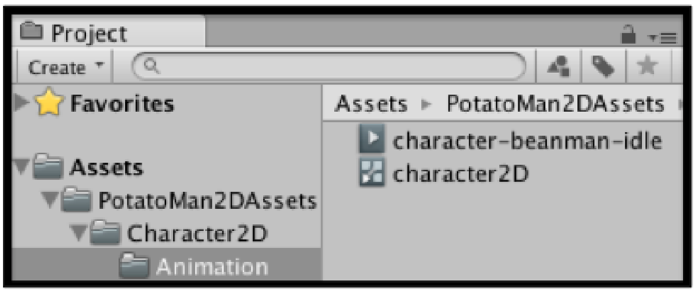
9. Select character2D in Hierarchy, then open Animation panel by **Window | Animation**, click Create button. Save the animation named Character2D in folder of **Character2D | Animation**.



10. Click dropdown menu in Animation panel, then click item menu [Create New Clip].

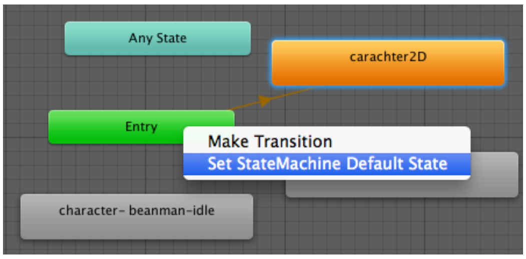


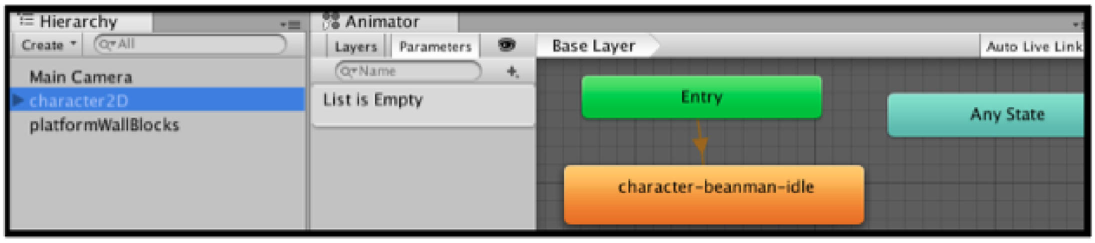
11. Save your new clip in **Character2D | Animation**, name it to be character- beanman-idle. In this stepm you have created an Animation clip for 'idle' character state (unmove).
12. Look at **Character2D | Animation** folder in Project panel, it should be 2 files: 1) Animation clip (character-beanman-idle) and 2) Animator controller from character2D (character2D).
    Note : Delete character2D clip because we don’t need this file anymore.



13. Select character2D in Hierarchy, open Animator panel (**Window | Animator**). You will see some State Machines to control animation from character. In this step, it has just one Animation clip (character-beanman-idle), so add clip character-beanman-idle at entry State Machine.

    Right click on Entry then select Set StateMachine Default State, direct it to clip character-beanman-idle.

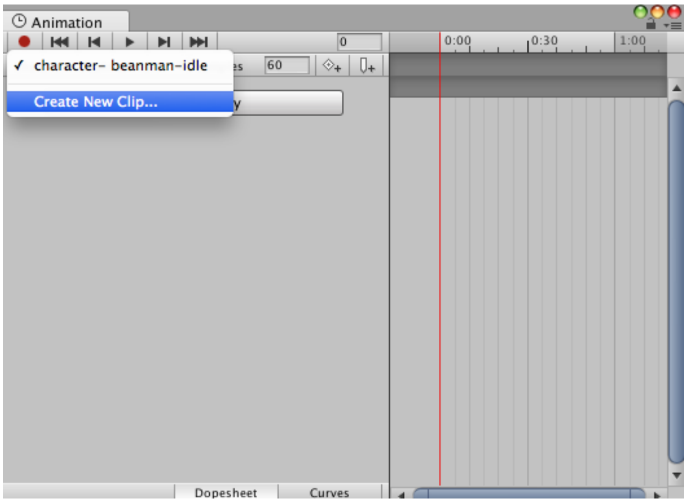




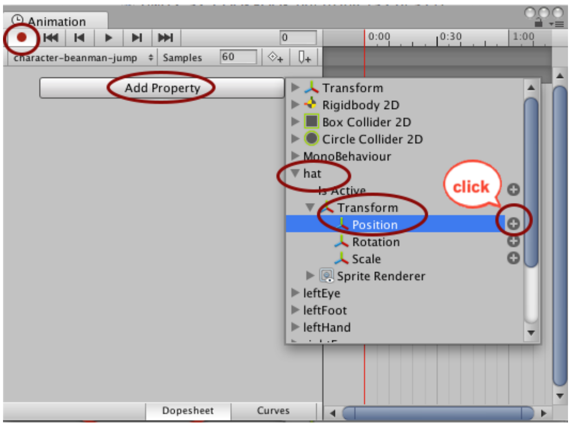
14. Run your project, you will see a character which is on 'idle' state, you can not move your character until you create another clip.

    Note: the character will drop to down because the garvity value is 1 in RigidBody 2D. The character will stop at platformWoodBlocks because they have Box Cllider 2D.

15. To create 'jump' Animation clip for hat animation, click the empty dropdown menu in Animation panel, then create a new clip at **character-beanman-jump** save in Animation folder.

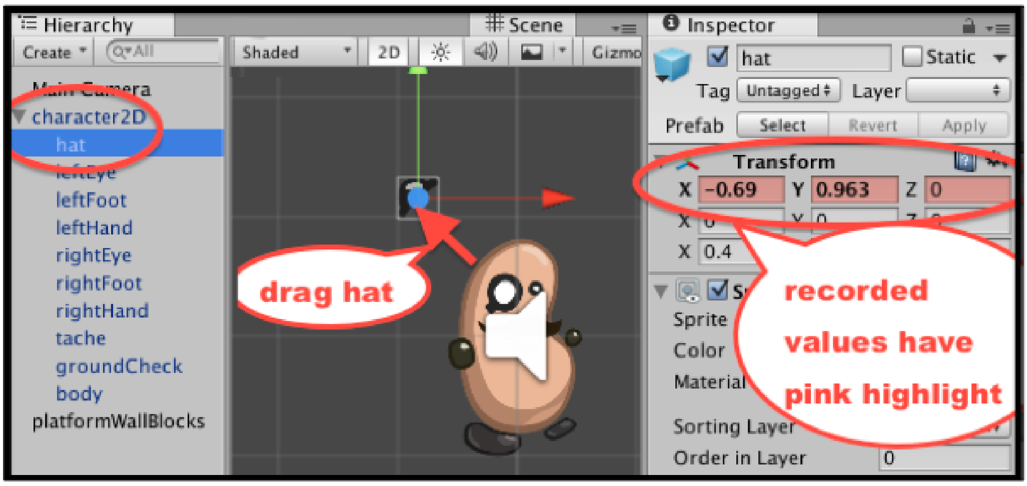


16. Click Add Property button, then select **Transform | Position** as hat child object by click '+' plus-sign button. In this step, we will do a recording of position (X, Y, Z) at GameObject hat in clip animation.

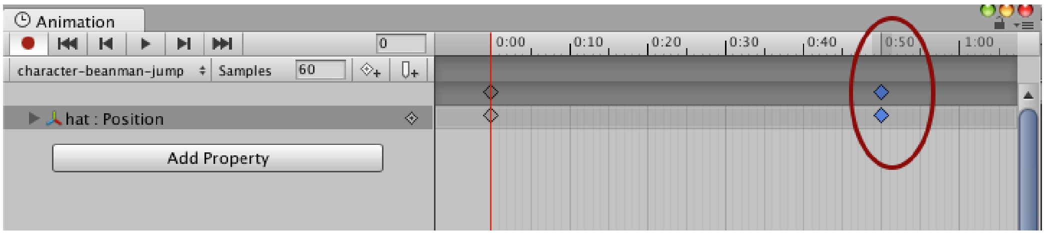


17. Look at 2 'keyframes' at 0.0 and at 1.0. It shown as diamonds in Timeline area at the right side of Animation panel.

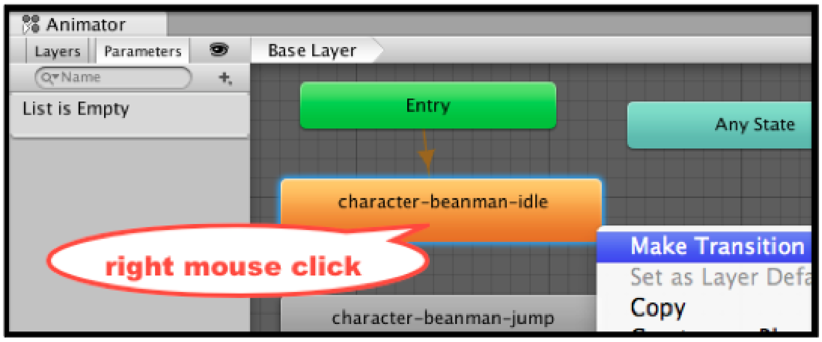
18. Select the first keyframe at (0,0) then in the **Scene panel** change the hat position to be upper left side position away from the head. Look at X,Y,Z value in inspector, the red background shows that any changes in Transform component is recorded in animation clip.



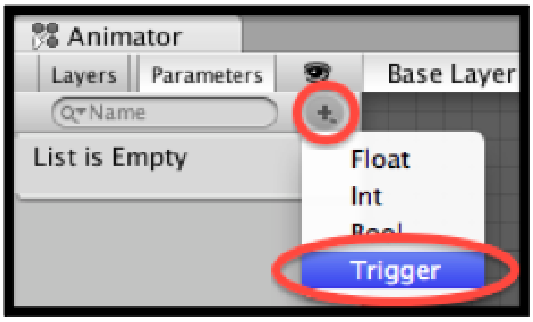
19. Because 1 second is too long for jump animation, drag the second keyframe to 0.5.



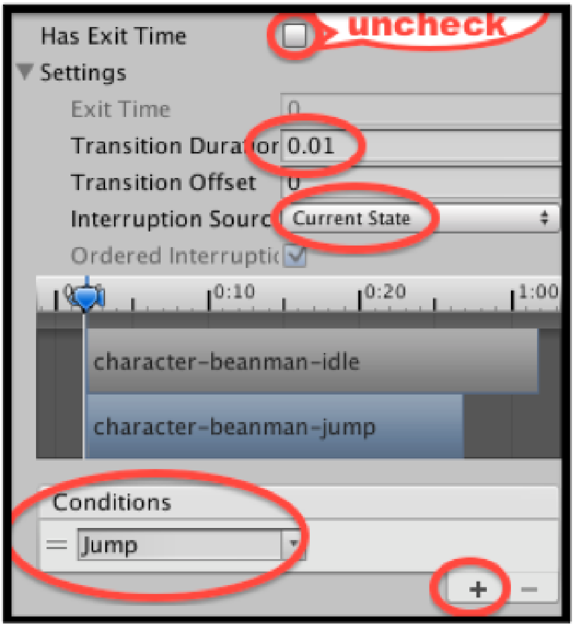
20. In Animator panel, select character-beanman-idle then make transition to state character-beanman-jump by right clicking and select Make Transition, then drag transition arrow to state character- beanman-jump.



21. Add Trigger parameter named 'Jump' by clicking add parameter plus-sign "+" at the upper left side of Animator panel, select Trigger then name Jump.



22. Make sure that PlayerControl script has been added to GameObject character2D.
23. In Animator Panel, set properties to determine when the character to make Transition from idle to jump by click Transition arrow, then change these 4 following things in Inspector panel, such:
    a. Has Exit Time: uncheck
    b. Transition Duration: 0.01
    c. Interruption State: Current State
    d. Conditions: Add Jump (click plus-sign '+' button at bottom)

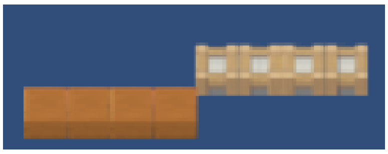


Has Exit Time □
Settings
Exit Time 0
Transition Duration 0.01
Transition Offset 0
Interruption Source Current State
Ordered Interruption ☑
Conditions
Jump
+ −

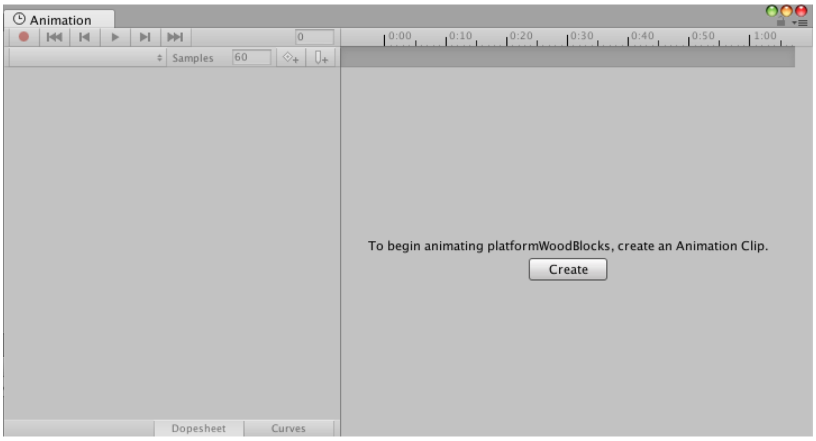
24. Run your project. When character landed, press space key to jump, you will see that the hat jumped away from the character's head, then be back slowly. In this step, we haven't create a transition to move from Jump state so the Animation clip will perform repeatedly (hat will always move away althoug the jumping activity has been stopped).
25. In Animator panel, select state character-beanman-jump then add Transition to bring back to state character-beanman-idle. Select Transition arrow in Inspector panel then change its properties
    a. Has Exit Time: check
    b. Exit time :0.5 (value must be the same as value in second keyfame at clip Jump animation)
    c. Transition Duration: 0.01
    d. Interruption State: Current State
26. Run your project. The hat will be only move once then be back to Idle state.

3. 3-Frame Animation Clip To Make Animation Runs Repeatedly

1. Continuing our second activity, create a new scene with different name from previous scene by **File | save scene as …** (example: 3FrameAnimation).
2. Drag sprite platformWoodBlocks from **Project | Sprites** to the scene. Change the position of GameObject to (-4, -5, 0).

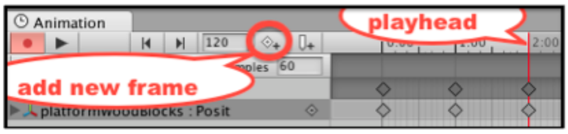


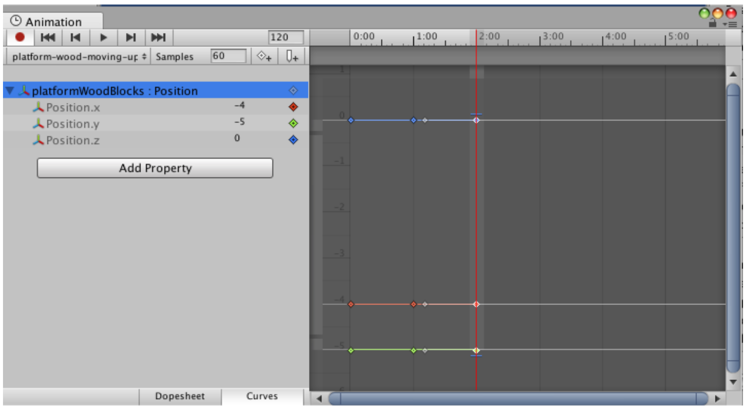
3. Add Box Collider 2D component to GameObject platformWoodBlocks so character2D may stand on this GameObject by **Add Component | Physics 2D | Box Collider 2D**.
4. Create a new folder named Animations to save animation clip dan controller.
5. Make sure that GameObject platformWoodBlocks is clicked in hierarchy then open Animation panel (**Window | Animation**).
6. Create a clip by click create button in Animation Pannel, then name it **platform-wood-moving-up-down** and save to Animations folder.



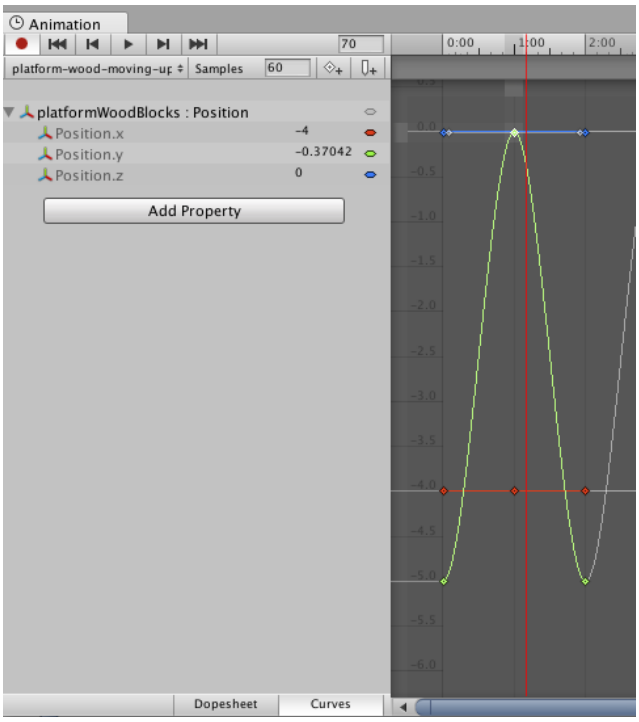
To begin animating platformWoodBlocks, create an Animation Clip.

7. Click Curve button in Animation Panel, then select Transform | Position, then click '+' sign. In this step, position change of (X, Y, Z) at GameObject platformWoodBlocks will be recorded.
8. You will see 2 'keyframes' at 0.0 and 1.0. The keyframe is represented by diamonds.
9. 3 keyframes is needed in this activity, to add 1 keyframe more, please click 2:00 at the Timeline then click diamond+ to create a new keyframe.





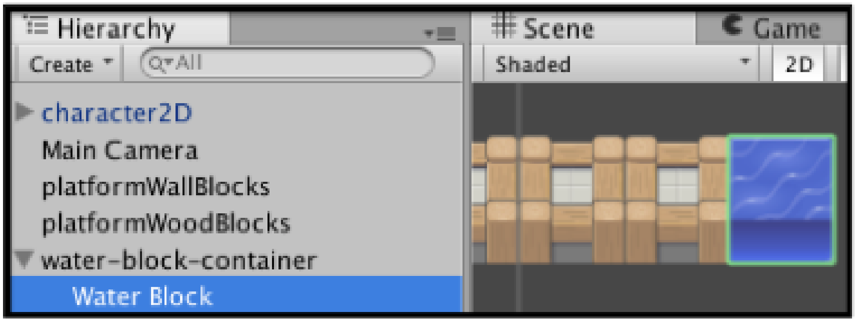
10. Keyframe pertama dan ketiga sudah benar nilai Y = -5 karena keyframe tersebut merekam/record tinggi dari wood platform. Akan tetapi kita mengubah keyframe yang di tengah sehingga merekam ketinggian wood platform ketika berada di puncak gerakannya. Dengan cara pilih keyframe yang kedua (pada timeline ke 1:00).
11. Setelah memilih keyframe ke 1:00 kemudian ubah nilai Y=0. Dan yang perlu diperhatikan nilai X,Y,Z pada inspector memiliki background merah yang menandakan Transform component sedang direkam pada animation clip.



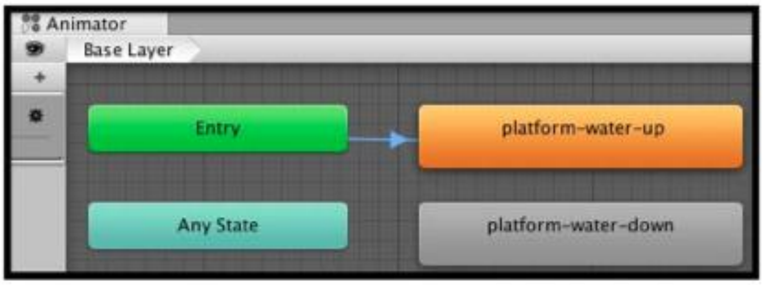
12. Run your project. The wood platform will move up and down slowly and repeatedly.

4. Create Animation from One State to Another State

1. Continouing our previous activity, create a new scene named MoveStateToState by **File | save scene as ….**
2. In hierarchy, create a new Empty GameObject named **water-block-container** with position (2.5, -4, 0) to make Water Block animation.
3. Drag sprite Water Block from **PotatoMan2DAssets | Sprites** to scene, place it to be child of GameObject water-block-container. Make sure GameObject Water Block position is (0, 0, 0) (sided with wall block platform)

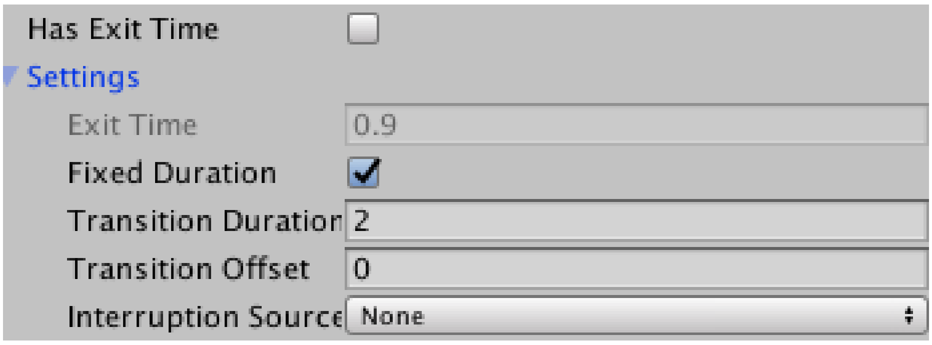


4. Add Box Collider 2D at GameObject Water Block, change the layer GameObject to be Ground so the character2D may jump up the water block platform.
5. Make sure GameObject Water Block has been selected in Hierarchy, open Animation Panel then create a new clip named **platform-water-up**. Save it to folder Animations.
6. Click Curves then add property then select **Transform | Position**.
7. Remove the second keyframe at 1:00 to activate the upstream of water block animation.
8. Create the second animation clip named **platform-water-down** then add property and select **Transform | Position**. Remove the second keyframe at 1:00.
9. Still on **platform-water-down,** set the timeline to 0:00 then change the value of Y=-5. It will make a downstream water block animation.
10. Open Animator panel, it will look like this:



11. Although 2 animation clips (states) have been created, It is only Up state that active. It is because when scene starts, (Entry state) object will go to state **platform-water-up,** but no transition to state **platform-water-down,** so GameObject Water Block will always be Up state.

12. To tackle the problem in number 11 above, it needs a transition to state platform-water-down by right clicking platform-water-up then select Make Transition and point to state platform-water-down.
13. Run your project. After 0.9th seconds, Water Block transition will go to Down state. To make downward animation when the player walks on Water Block, it needs Trigger. Create a Trigger named Fall, by select Parameter tab in Animator Panel, then click button '+', select Trigger then select Fall.
14. Do this following steps to set the Trigger:
    a. In Animator panel, select Transition.
    b. Has Exit Time: uncheck
    c. In Inspector panel, drag Transition end time to 2:00 seconds (Water Block will make a Transition slowly to Down state for 2 seconds)
    d. Conditions: Add Fall (click plus-sign '+' button at bottom)



Has Exit Time ☐

Settings

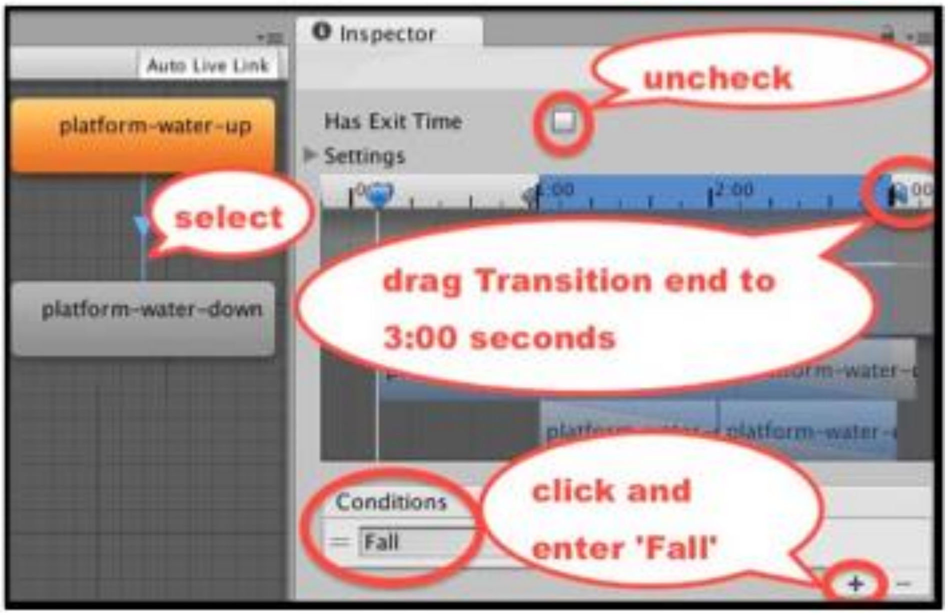
Exit Time 0.9

Fixed Duration ☑

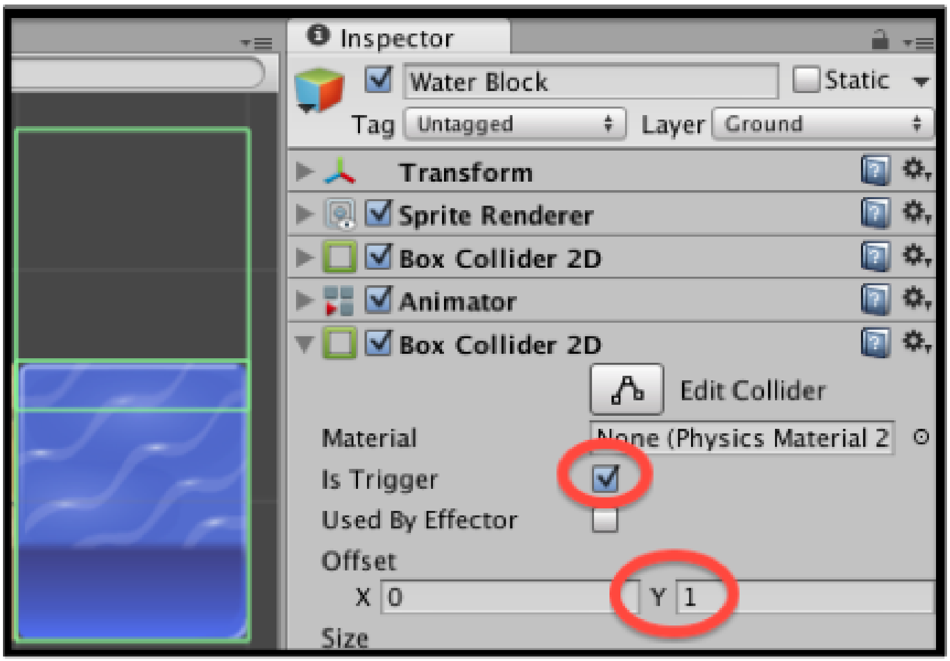
Transition Duration 2

Transition Offset 0

Interruption Source None



15. Collider trigger must be added to Water block, also a C# script to deliver Animator Controller Trigger when player enters collider. Make sure GameObject Water Block has been selected, add the second Box Collider 2D with Y-Offset=1 then tick Trigger checkbox.



Inspector

☑ Water Block □ Static

Tag Untagged Layer Ground

Transform

☑ Sprite Renderer

☑ Box Collider 2D

☑ Animator

☑ Box Collider 2D

Edit Collider

Material None (Physics Material 2

Is Trigger ☑

Used By Effector □

Offset

X 0 Y 1

Size

16. Create a new C# script named WaterBlock with this following code, then add this script to GameObject Water Block.

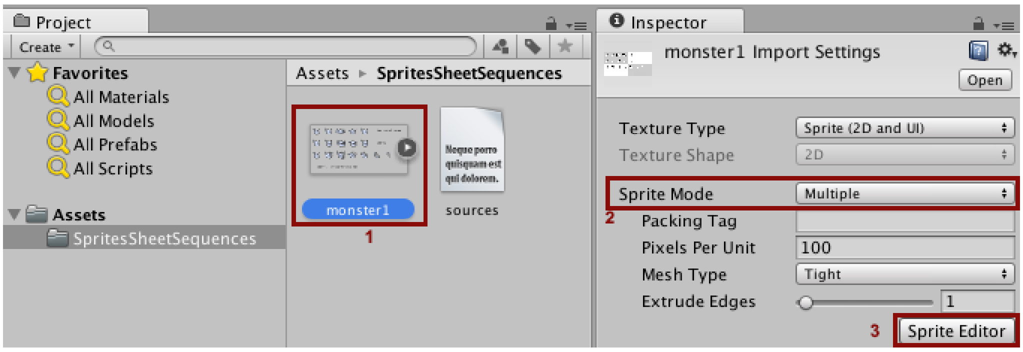
```
using UnityEngine;
using System.Collections;

public class WaterBlock : MonoBehaviour {
    private Animator animatorController;
    void Start(){
        animatorController = GetComponent<Animator>();
    }
    void OnTriggerEnter2D(Collider2D hit){
        if(hit.CompareTag("Player")){
            animatorController.SetTrigger("Fall");
        }
    }
}
```

17. Duplicate GameObject water-block-container 6 times with position of X raises 1 point to each GameObject, for example 3.5, 4.5, 5.5, etc.
18. Run your project. Look at when the player passes the water block, the water block will make a downward, and the player keeps going.

5. Animation Clip from Sprite Sheet Sequences

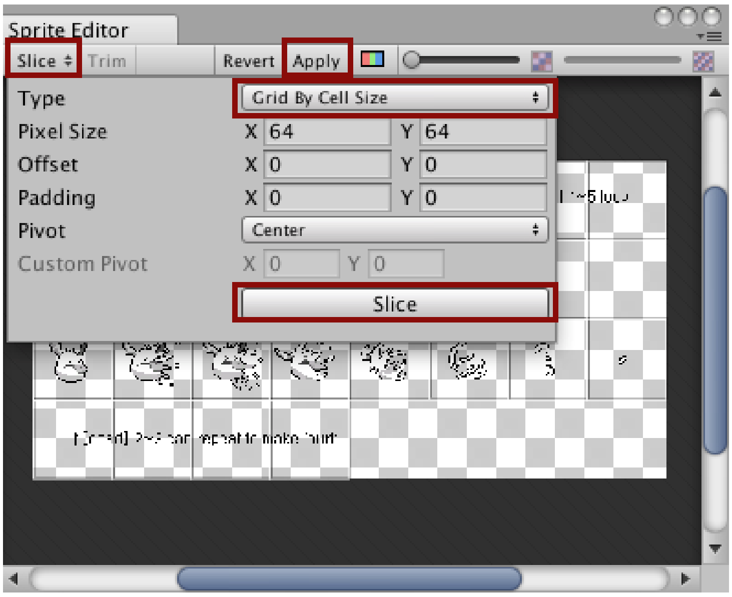
1. Create a new 2D Unity project.
2. Drag folder SpritesSheetSequences into asset.
3. Select image monster1 at Project panel, change sprite mode to Multiple in inspector, then open Sprite Editor panel by click button Sprite Editor.



Inspector: monster1 Import Settings

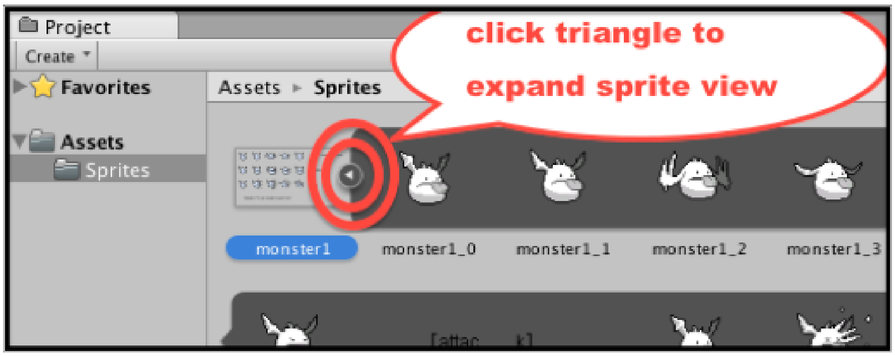
Texture Type: Sprite (2D and UI)
Texture Shape: 2D
Sprite Mode: Multiple
Packing Tag:
Pixels Per Unit: 100
Mesh Type: Tight
Extrude Edges: 1

4. In **Sprite Editor,** open **Slice** at dropdown dialog, set Type=**Grid**, then change the grid **Pixel Size** to be 64x64, then click button Slice. Click button **Apply** which is on **Sprite Editor** panel.



Type: Grid By Cell Size
Pixel Size: X 64 Y 64
Offset: X 0 Y 0
Padding: X 0 Y 0
Pivot: Center
Custom Pivot: X 0 Y 0

5. In Project panel, click expand triangle button at image monster1.



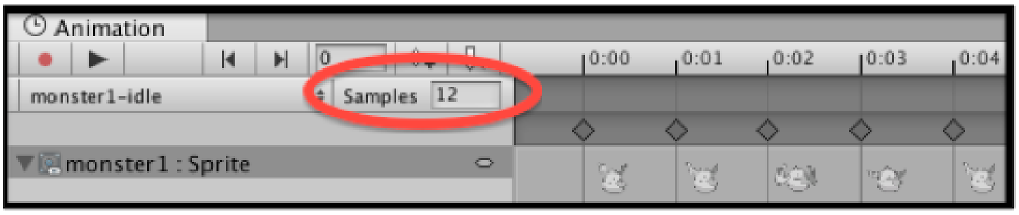
6. Create a new folder named Animation.

7. In Animation folder, create **Animator Controller** named **monster-animator** by click Create | Animator Controller.

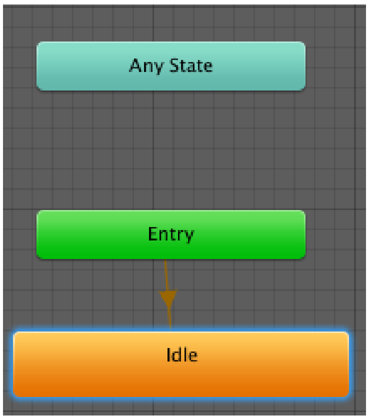
8. In the scene, create a new Empty GameObject named **monster1** then set position (X, Y, Z) to (0, 0, 0). Drag **monster-animator** to GameObject.

9. At **Hierarchy,** select GameObject **monster1**, then open **Animation** panel (at Window | Animation), then create **Animation Clip** named **Idle (**by click button Create in Animation panel**)**.

10. In Project Panel, select monster1 then drag the first of 5th frame (0th – 4th frame) one by one to Animation Panel. Set the Samples value to 12 which means this animation is used to run 12 frames per second.



11. Look at this state-chart, it shows that the default size named idle.



12. Run your project, it shows that GameObject monster1 runs on idle state. It happens because sprite is in small size, so Camera size can be smaller.

A. Assignment

1. Download the game assets from this link (https://www.assetstore.unity3d.com/en/#!/content/11228), give review and analysis what kind of steps to develop that game.

--- Good Luck ---