BINUS UNIVERSITY

People
Innovation
Excellence

Course : 2D Game Programming

Effective Period : September 2016

2D Game Programing
LAB 01

BINUS UNIVERSITY

People
Innovation
Excellence

Acknowledgement

These slides have been adapted from:

Pereira, V. (2014). Learning Unity 2D Game Development by Example, Packt Publishing, Inc. San Francisco. ISBN: 9781783559046

Chapter 4

BINUS UNIVERSITY

Learning Objectives

LO 1 : Create 2D game for PC platform
LO 3 : Design 2D game for PC platform

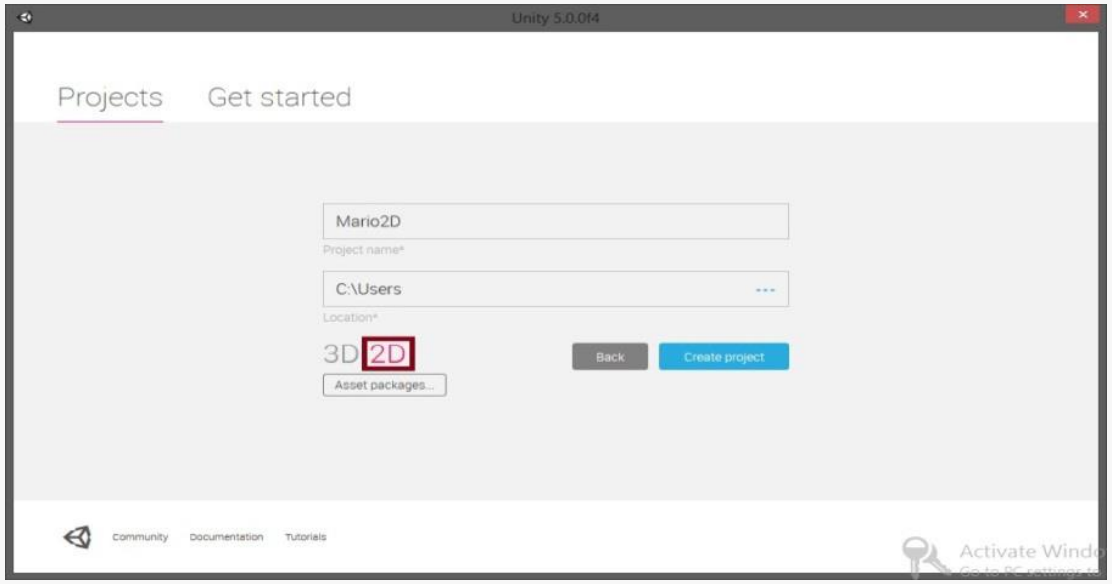
People
Innovation
Excellence

BINUS UNIVERSITY

People
Innovation
Excellence

Let's start

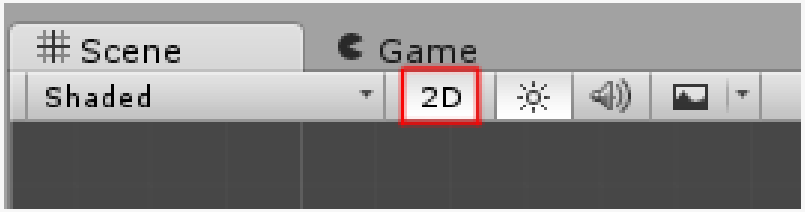
Open unity and pick 2D mode



People
Innovation
Excellence

2D view

Make sure your view is in 2D and the camera has orthographic projections.

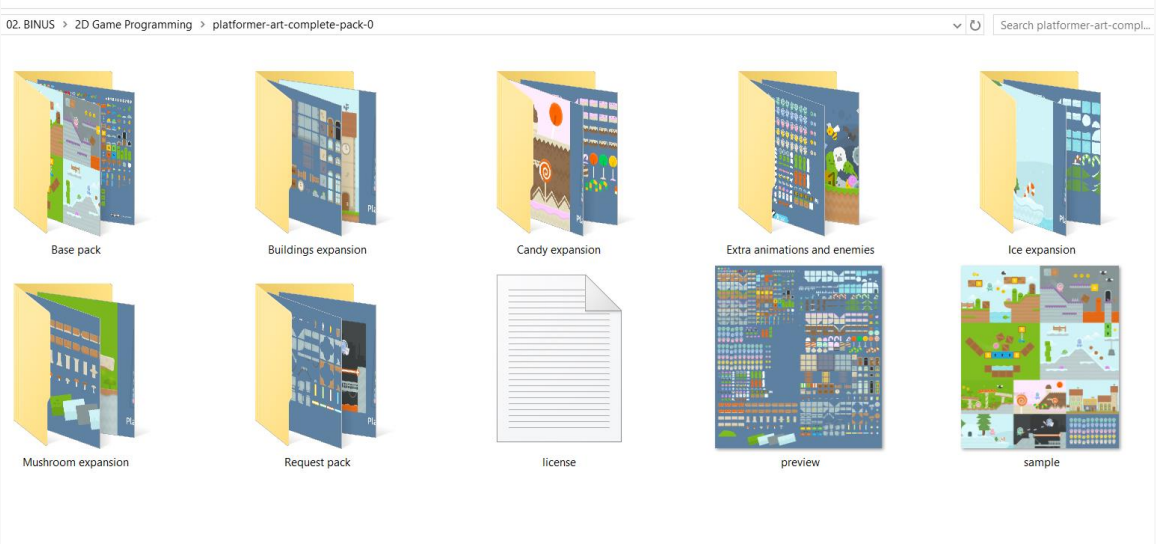


Click the main camera and check the inspector

People
Innovation
Excellence

Check out the files

Find platformer-art-complete-pack-0.zip and extract it.



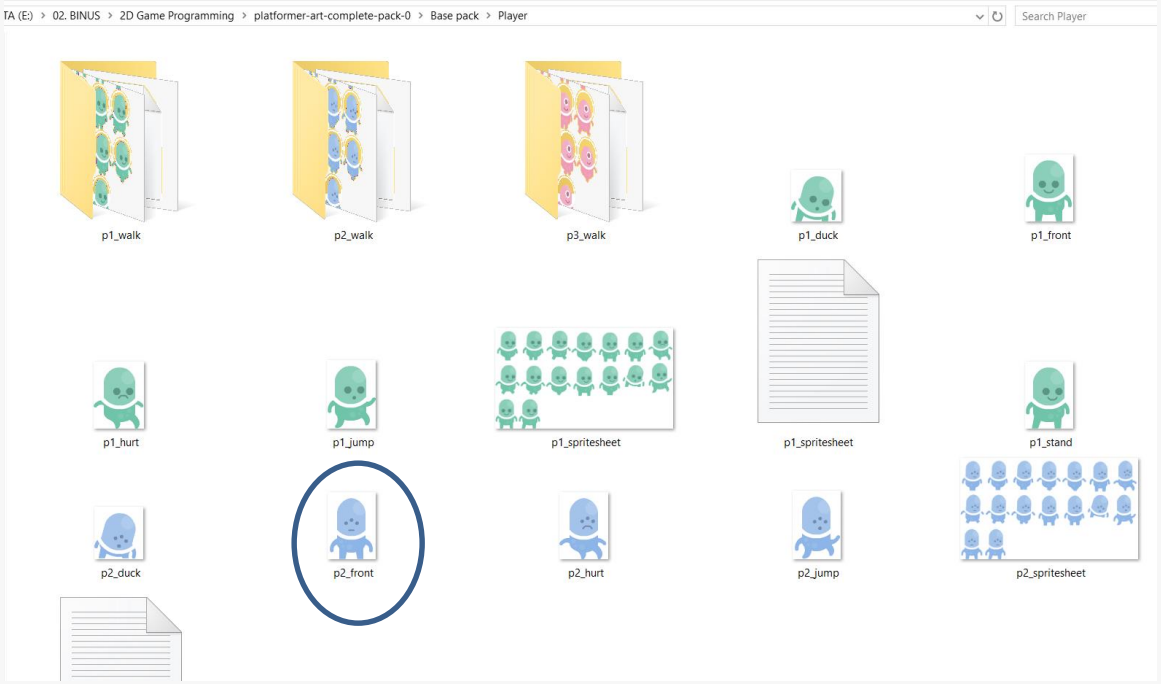
You can download here http://kenney.nl/assets/platformer-art-deluxe

BINUS UNIVERSITY

People
Innovation
Excellence

Opening player

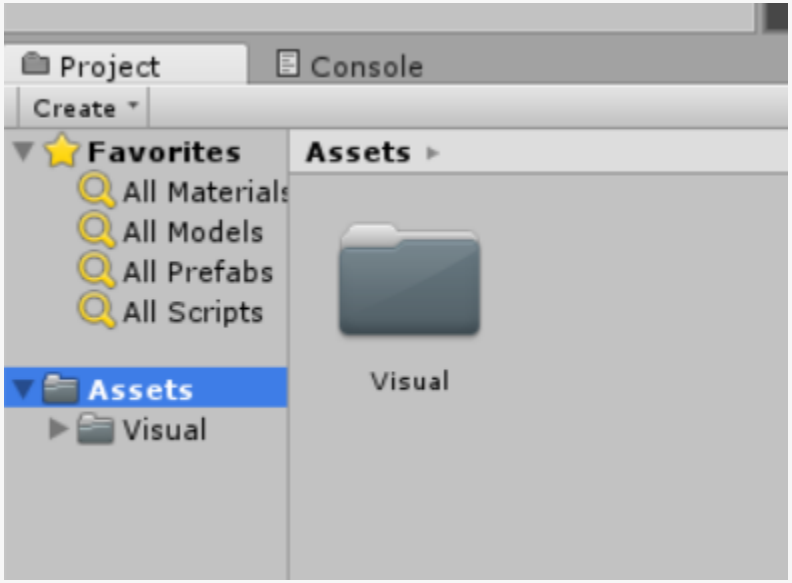
Search for p2_front

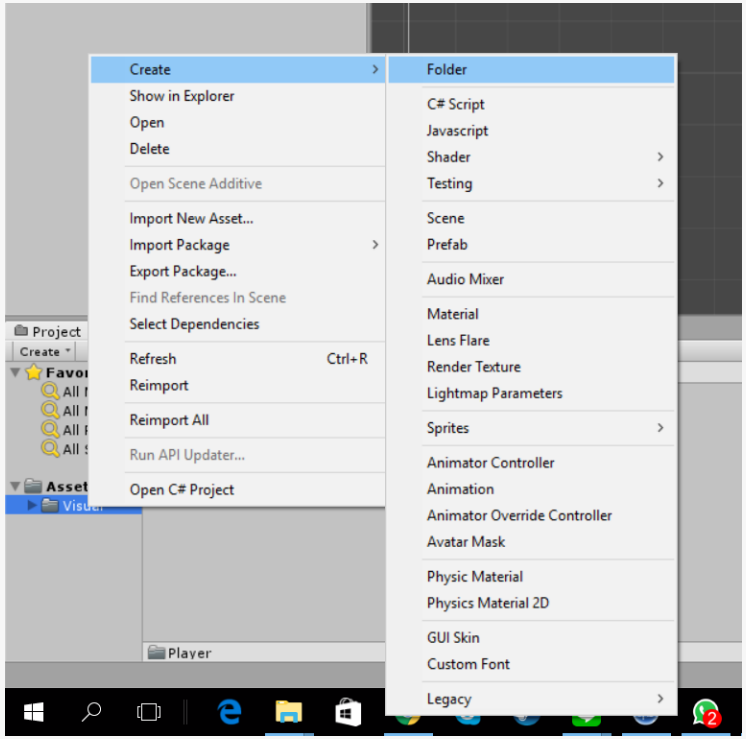


BINUS UNIVERSITY

People
Innovation
Excellence

Create folder visual and player within it



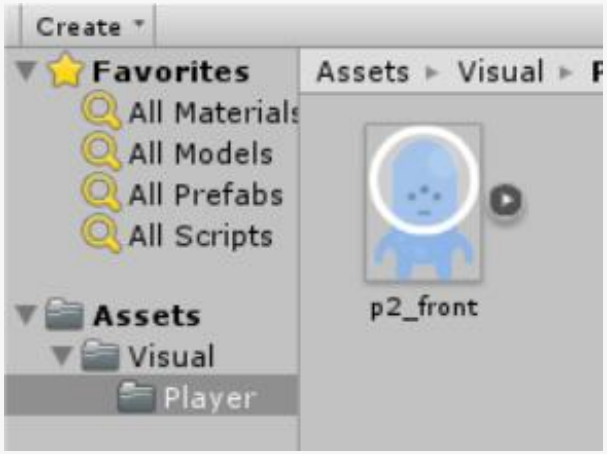


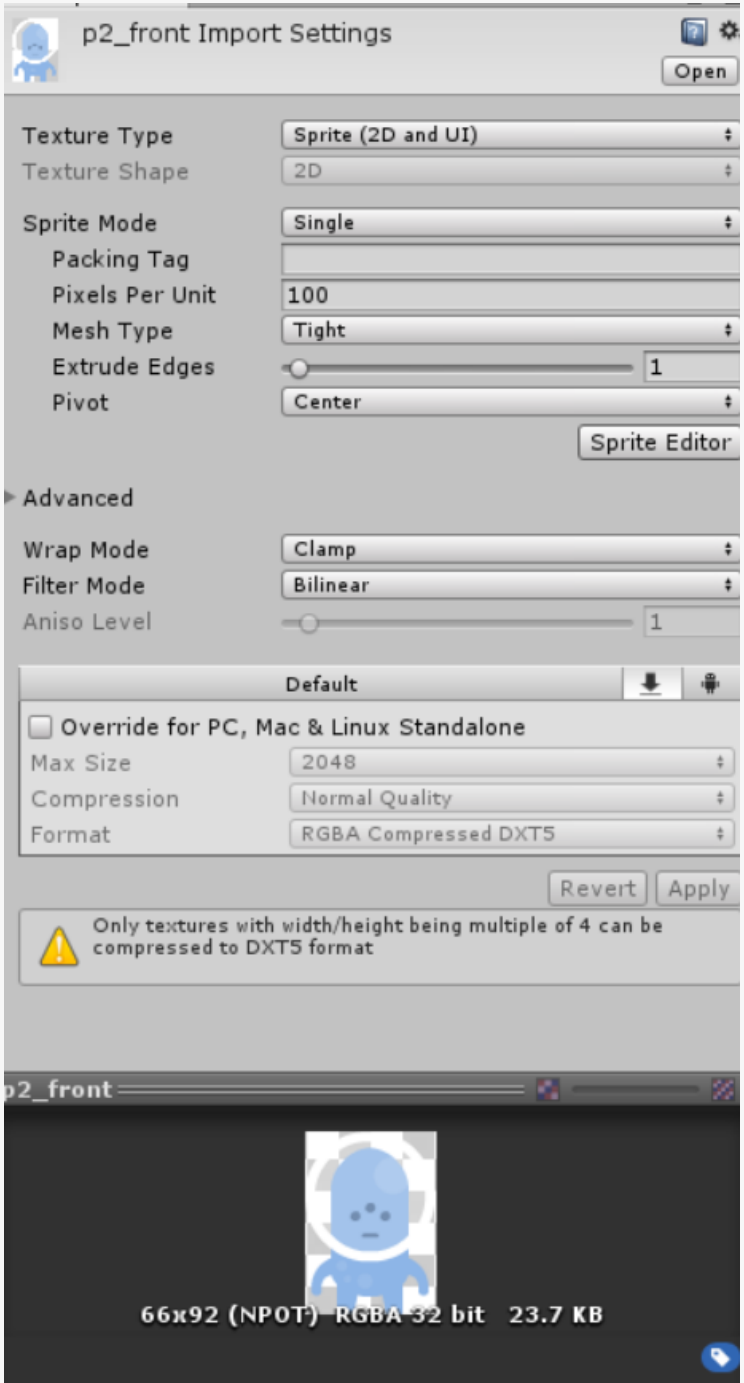
BINUS UNIVERSITY

People
Innovation
Excellence

Move the image

Drag and drop the image and check out the inspector





BINUS UNIVERSITY

People
Innovation
Excellence

What's in the inspector?

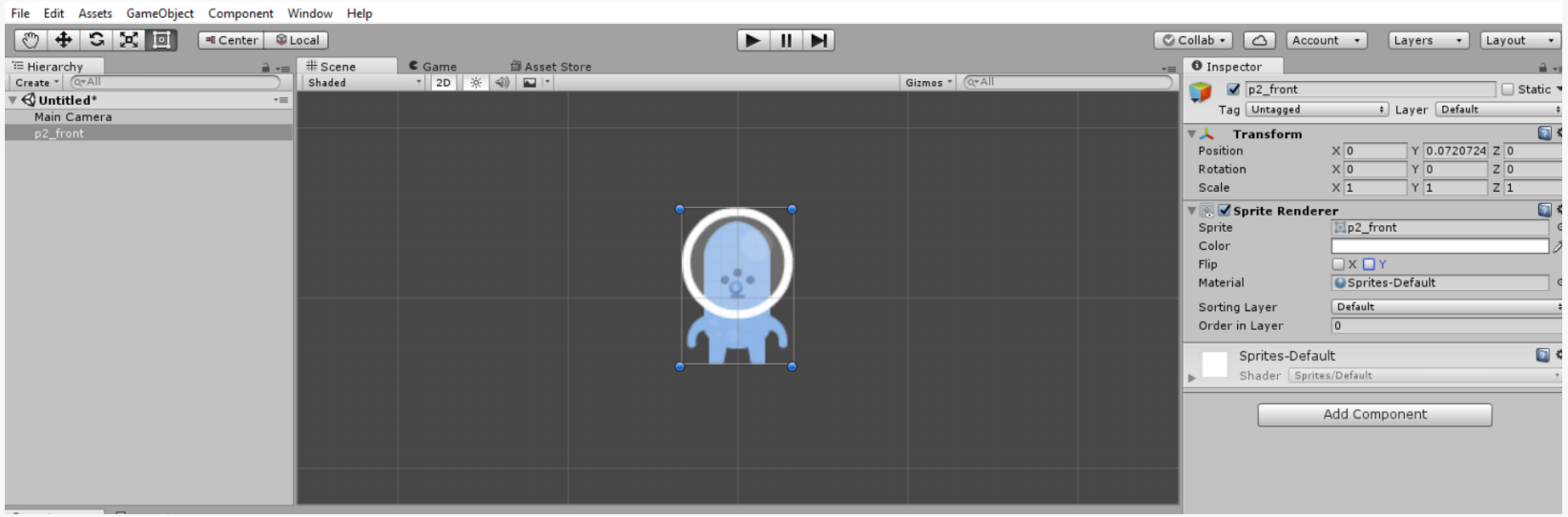
- **Sprite Mode**: This mode consists of two options Single and Multiple. Single should be selected when the image contains only a single object or character that will be used as a single sprite. Multiple, instead, it should be selected when multiple elements are contained within the same image .These may include different variations of the same object or its animation sheet.
- **Packing Tag**: This is an optional variable used to specify the name of the Sprite Sheet in which this texture will be packed. This is useful when we need to optimize our game.
- **Pixels Per Unit**: This controls the scale of the sprite. This variable defines how many pixels correspond to one world space unit. The default value for this is 100.
- **Pivot**: This allows us to change the pivot point for our sprite, which, by default, is set to Center. When required, you can change it to one of the other predefined points or place it in a custom position by selecting Custom.

BINUS UNIVERSITY

People
Innovation
Excellence

Move it to the game scene

Since we already have the p2_front sprite selected, let's drag it into the **Hierarchy** panel.

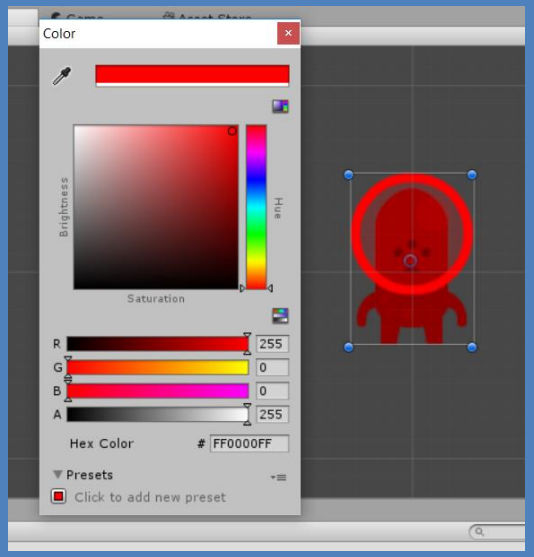


When we add a sprite in our scene, a game object is created with a Sprite Renderer component attached to it. This component is responsible for rendering a Sprite in the game; without it, the game object would be empty.

BINUS UNIVERSITY

People
Innovation
Excellence

Let's try out some variable



The **Color** variable controls the color of the **Sprite** along with its alpha channel.

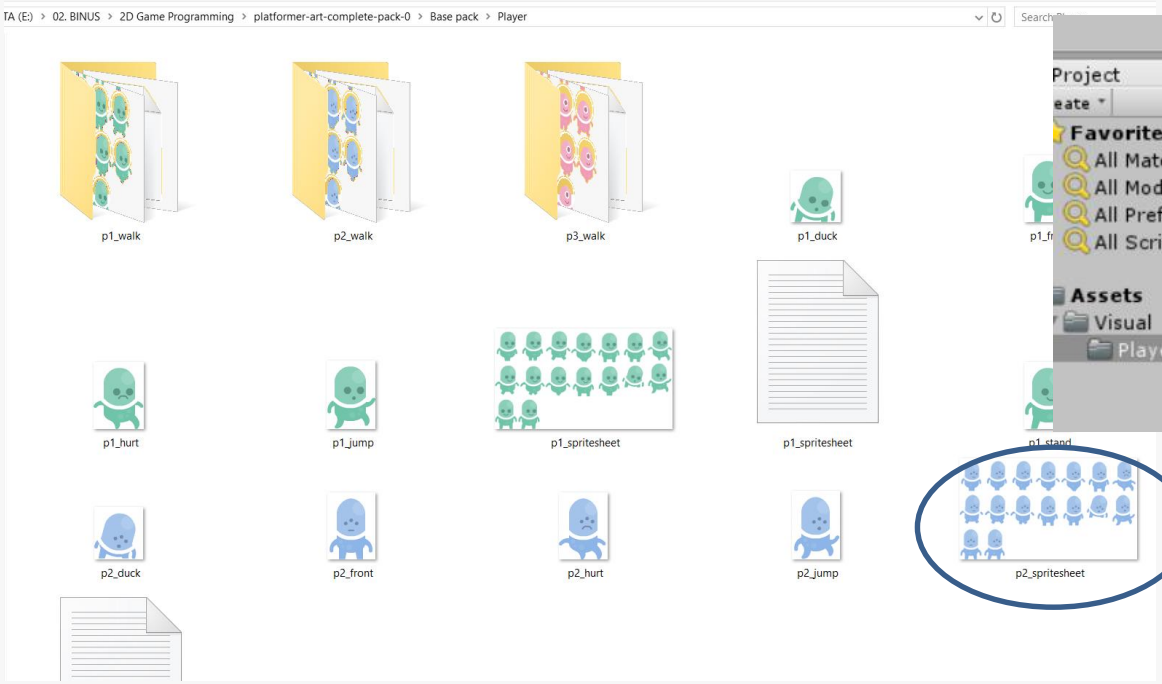
The **Material** variable stores information about the material of the sprite. By default, it is set to Default Sprite Material. Usually, we don't want to change this during 2D game development; however, it may be necessary in particular cases, for instance when the sprite needs to be affected by lights.

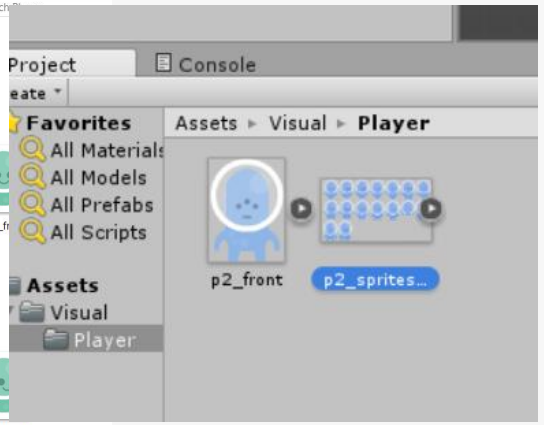
Then, the **Sorting Layer** and **Order In Layer** variables are used to define the order of visualization of the sprites in the scene.

People
Innovation
Excellence

The sprite editor

Move p2_spritesheet to the player folder



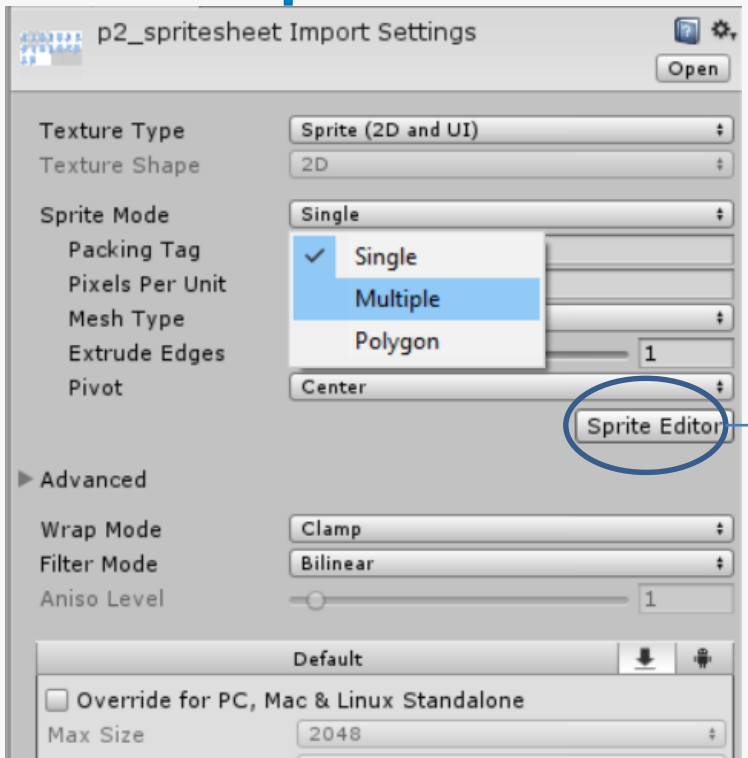


BINUS UNIVERSITY

People
Innovation
Excellence

The sprite editor

Edit the sprite mode before open the sprite editor

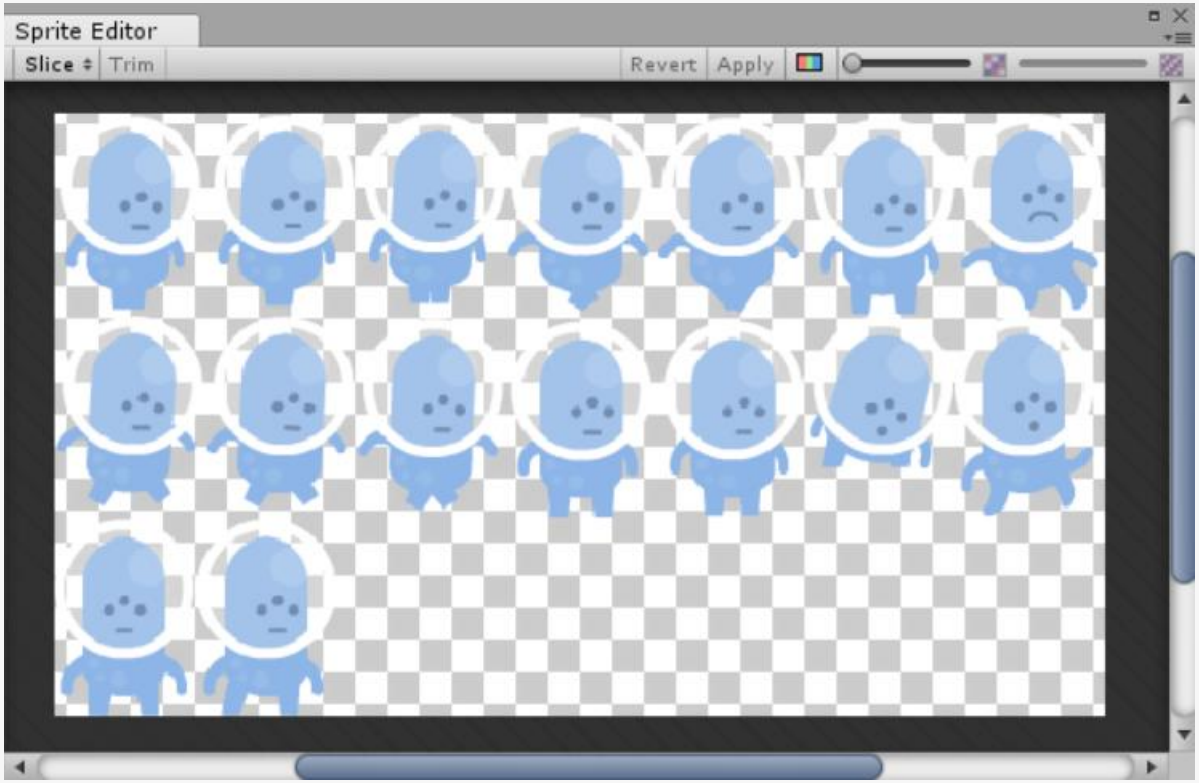


Press sprite editor after changing Sprite mode

BINUS UNIVERSITY

People
Innovation
Excellence

The Sprite Editor

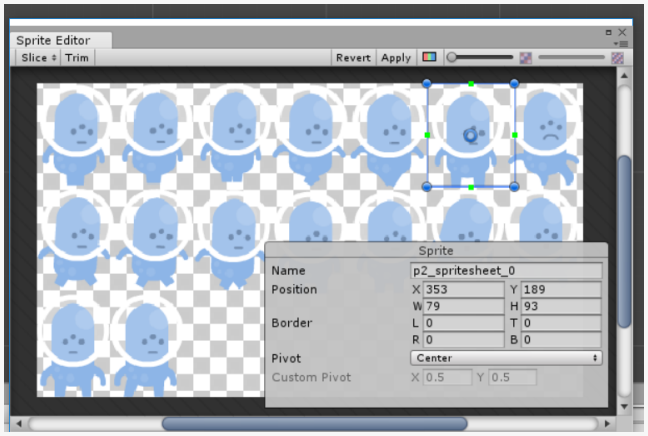


BINUS UNIVERSITY

People
Innovation
Excellence

The Sprite Editor

Our goal is to slice all the single positions of the character in the image, so that we can use them as an individual sprites in our scene.



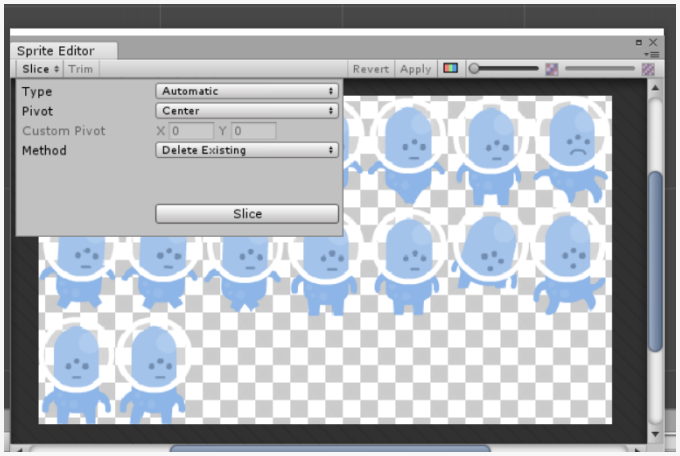
Click and drag: This allows you to simply click and drag over the desired elements to create rectangular selections that will define each sprite. You can change each selection as preferred. You are able to change the position by dragging the rectangle, its size by clicking on the corners of each rectangle, and the **Pivot** point by clicking and dragging it. Furthermore, clicking on the trim button in the sprite window will change the size of the rectangle to fit the selected sprite.

BINUS UNIVERSITY

People
Innovation
Excellence

The Sprite Editor

Our goal is to slice all the single positions of the character in the image, so that we can use them as an individual sprites in our scene.



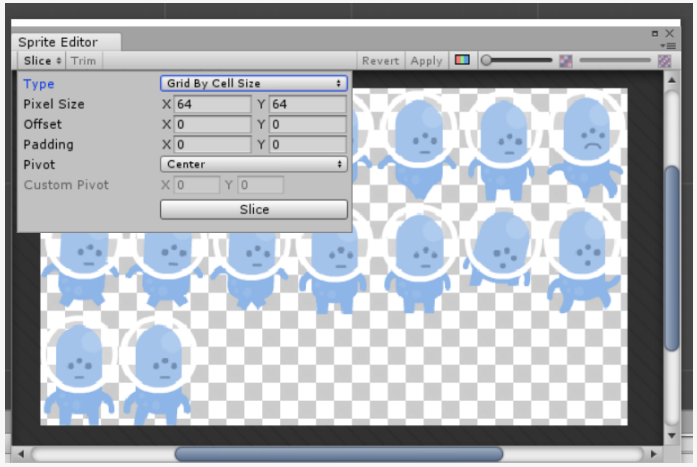
Automatic: When using the automatic method, Unity will detect each sprite and draw a trimmed rectangle around it. With Automatic selected, you can also change the **Pivot** position for each sprite. We can also select a **Method** to tell Unity what should happen to the sprites that are already defined. The Delete Existing method deletes all the previous selections and creates new ones from scratch. The Smart method attempts to create new selections for undefined sprites, while adjusting them to fit with the older selections. Finally, the Safe method adds new selections over the previous ones without changing them.

BINUS UNIVERSITY

People
Innovation
Excellence

The Sprite Editor

Our goal is to slice all the single positions of the character in the image, so that we can use them as an individual sprites in our scene.

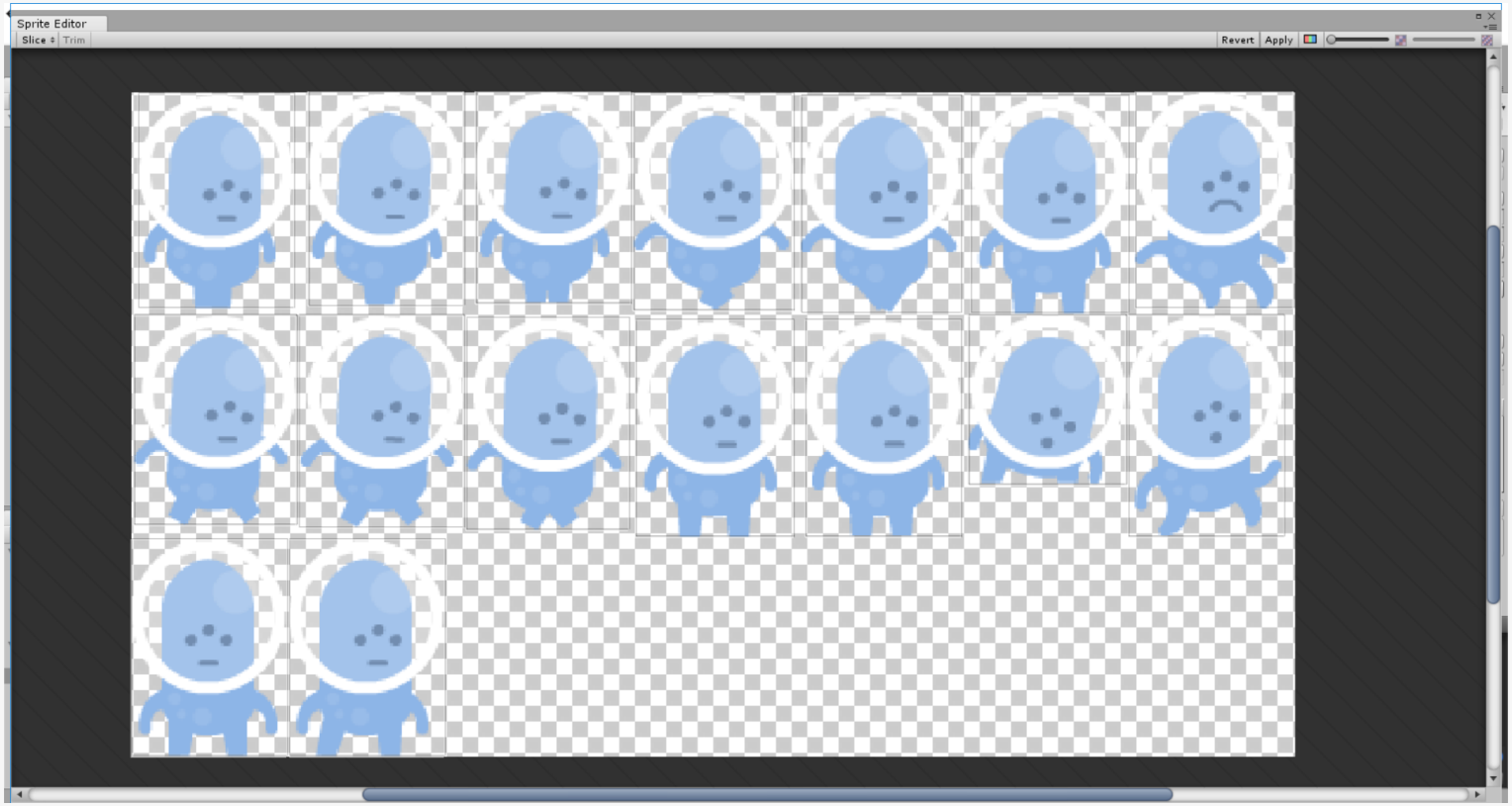


Grid: This allows you to create equal size selections for all the sprites in the image. Once we have set Grid as slicing type, then we will be able to change the size used for the slicing, and eventually adjust the position of the **Pivot** point for each sprite.

BINUS UNIVERSITY

People
Innovation
Excellence

USE THE AUTOMATIC OPTION



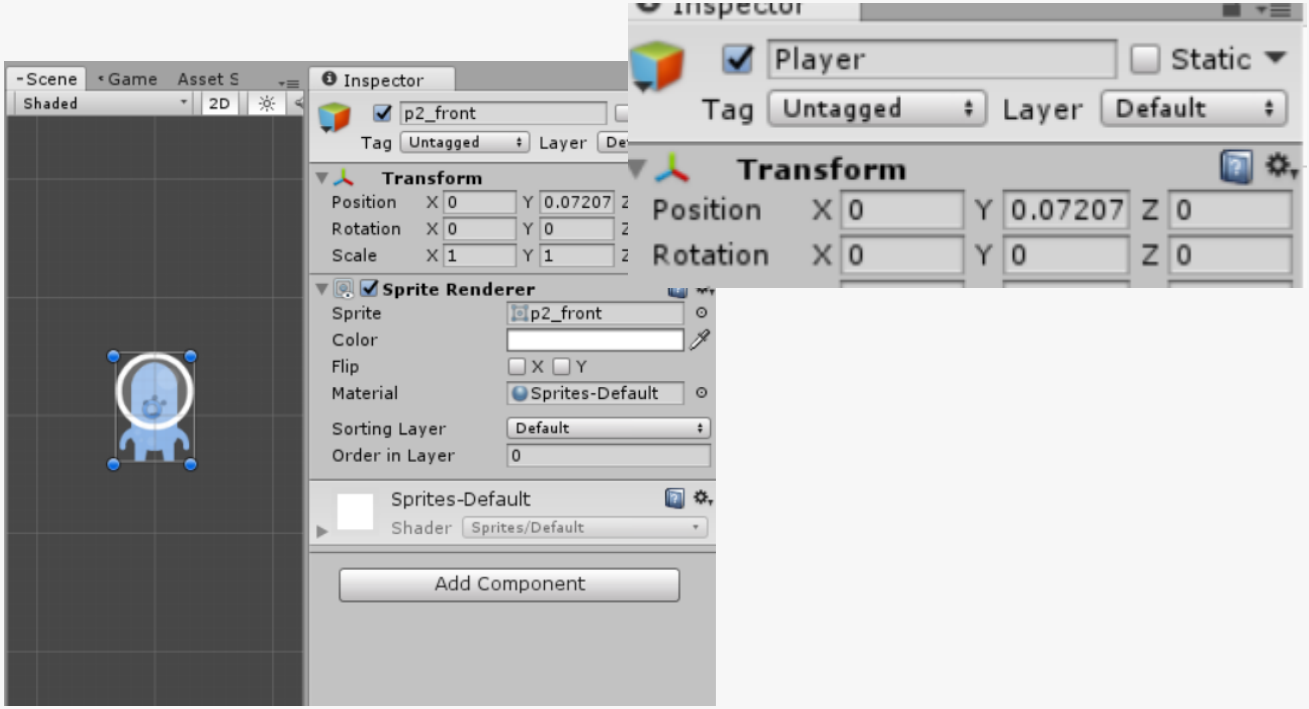
Don’t forget to press apply...

BINUS UNIVERSITY

People
Innovation
Excellence

Let's move

Rename the p2_front to player

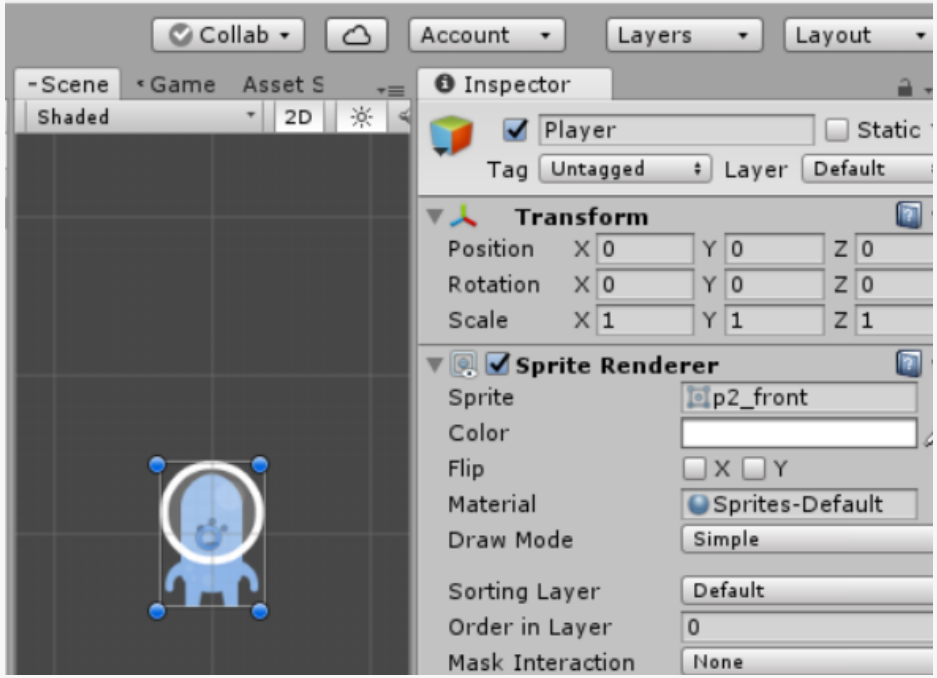


BINUS UNIVERSITY

People
Innovation
Excellence

Let's move

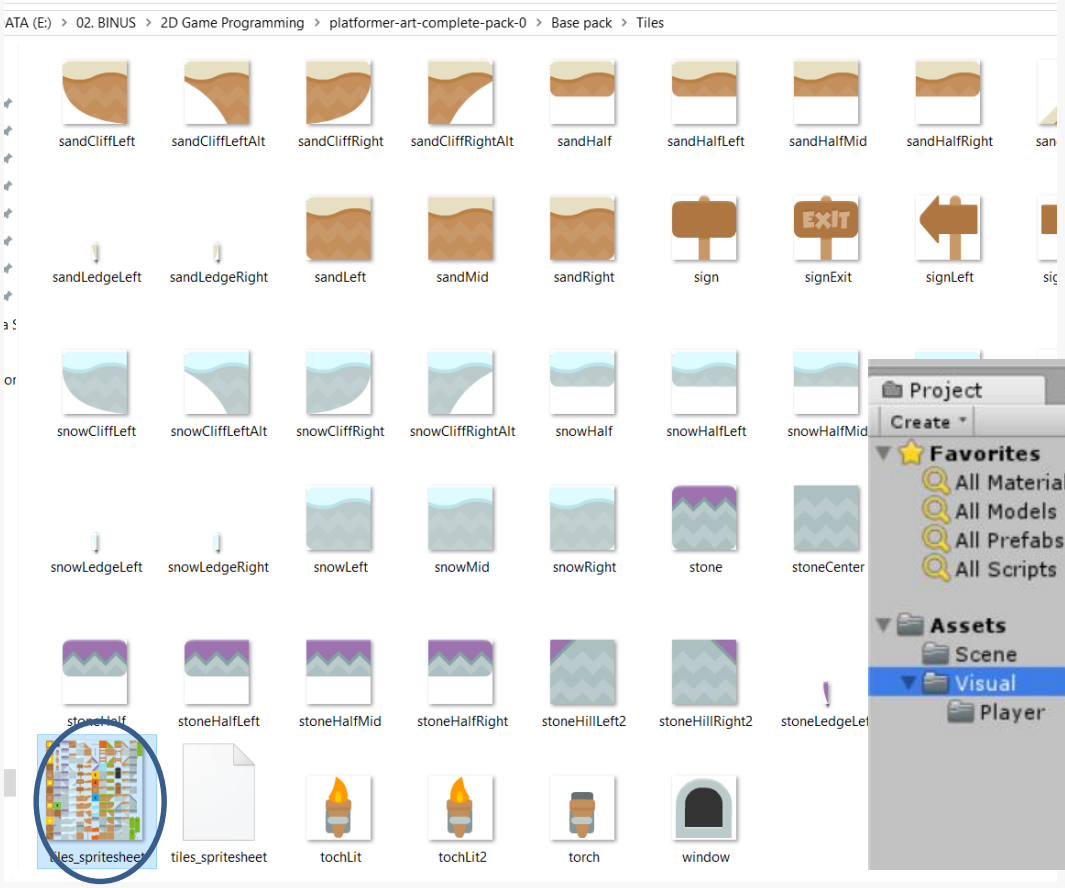
Make sure position is 0,0,0

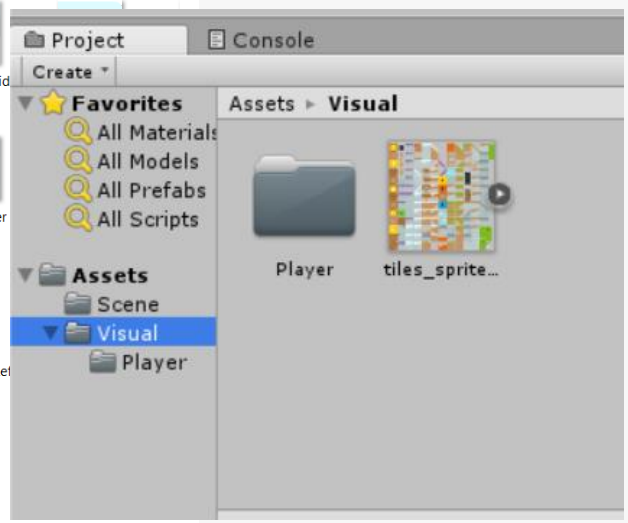


BINUS UNIVERSITY

People
Innovation
Excellence

Let's move





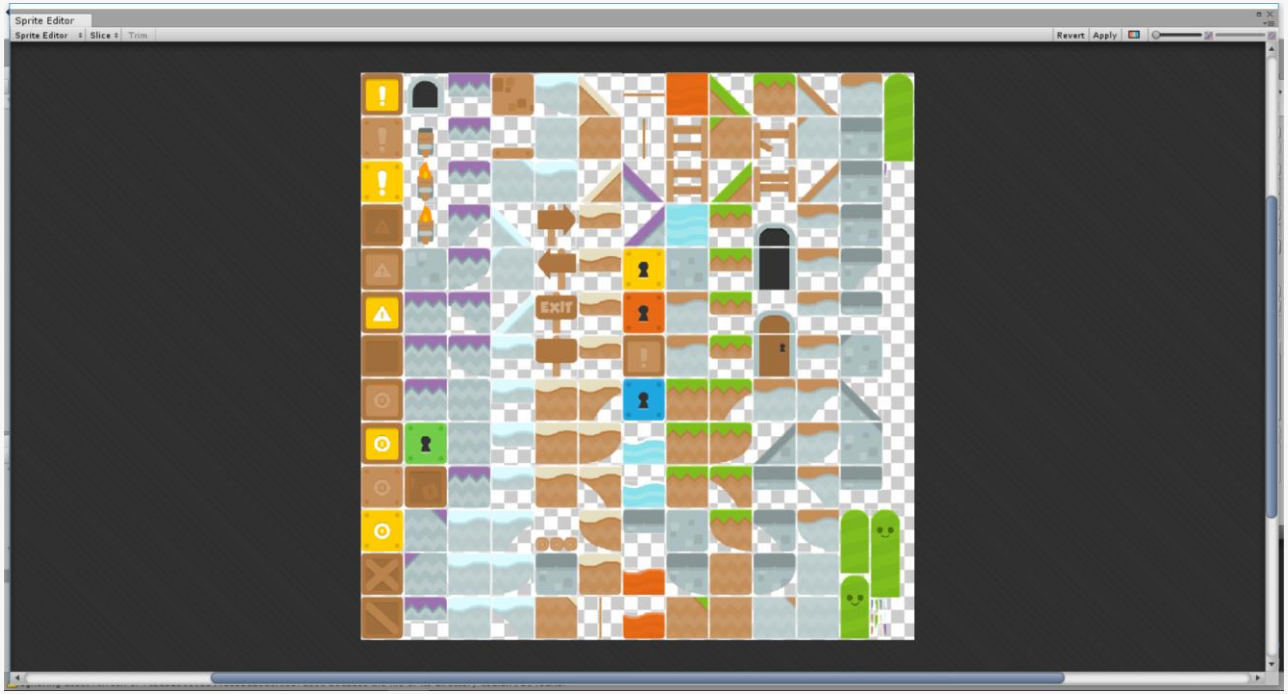
Move the tiles sprite sheet to the project folder

BINUS UNIVERSITY

People
Innovation
Excellence

A little thing you need to do

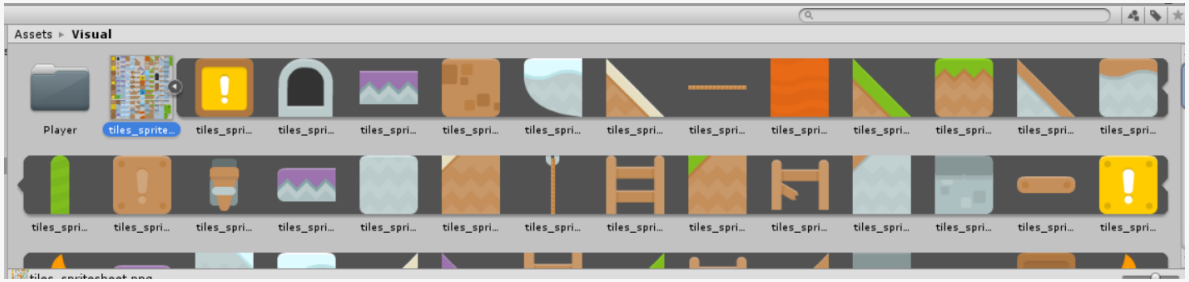
Use automatic to splice the image using tile editor

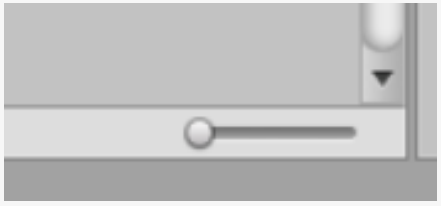


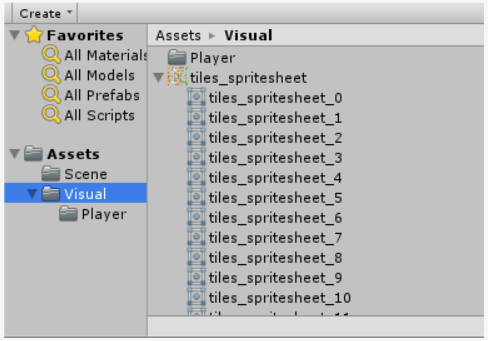
People
Innovation
Excellence

It's spliced!!

Use bottom right slider to tiggle file view





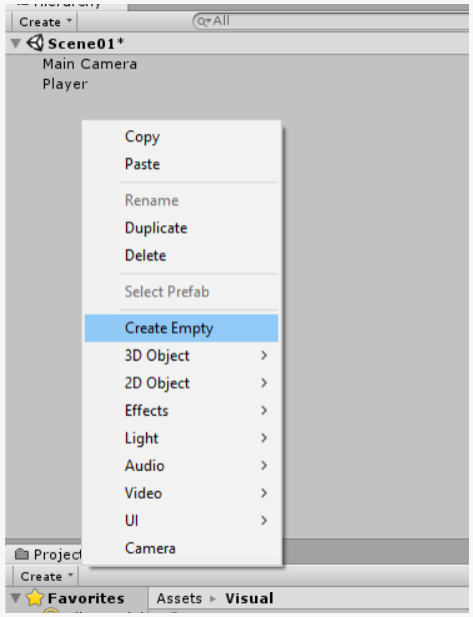


BINUS UNIVERSITY

People
Innovation
Excellence

Create a platform

Create an empty game project that will house the platform and name it “Floor”

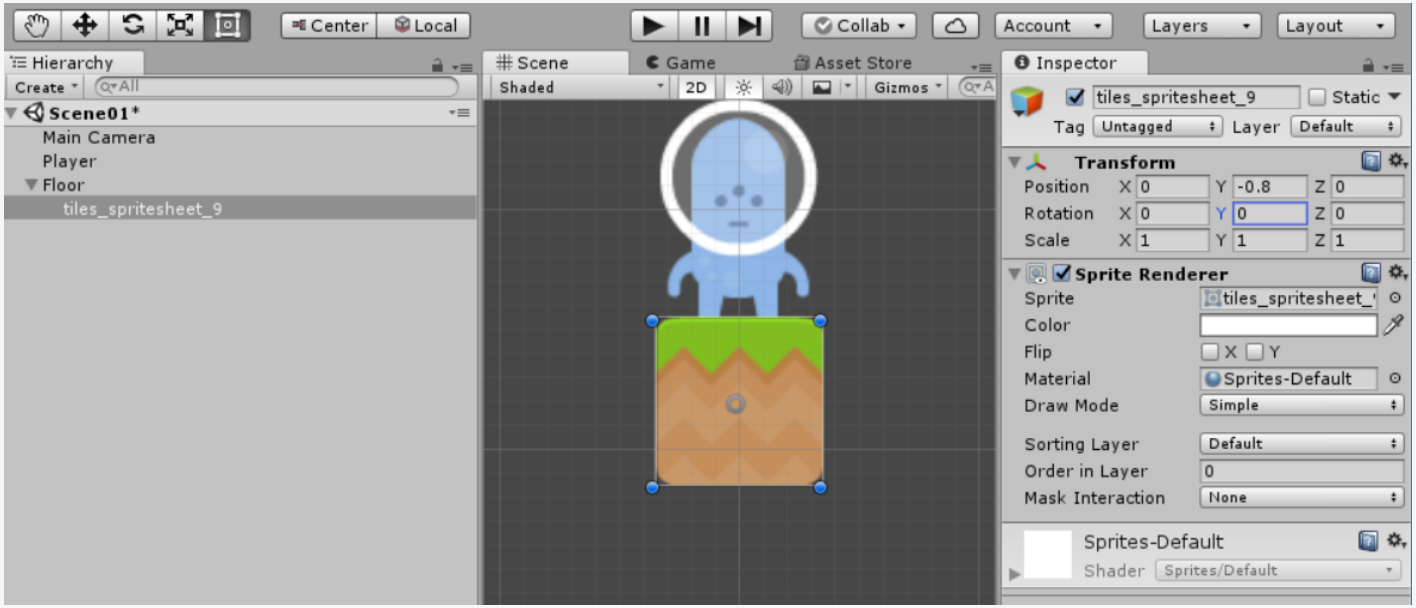


BINUS UNIVERSITY

People
Innovation
Excellence

Visualize the platform

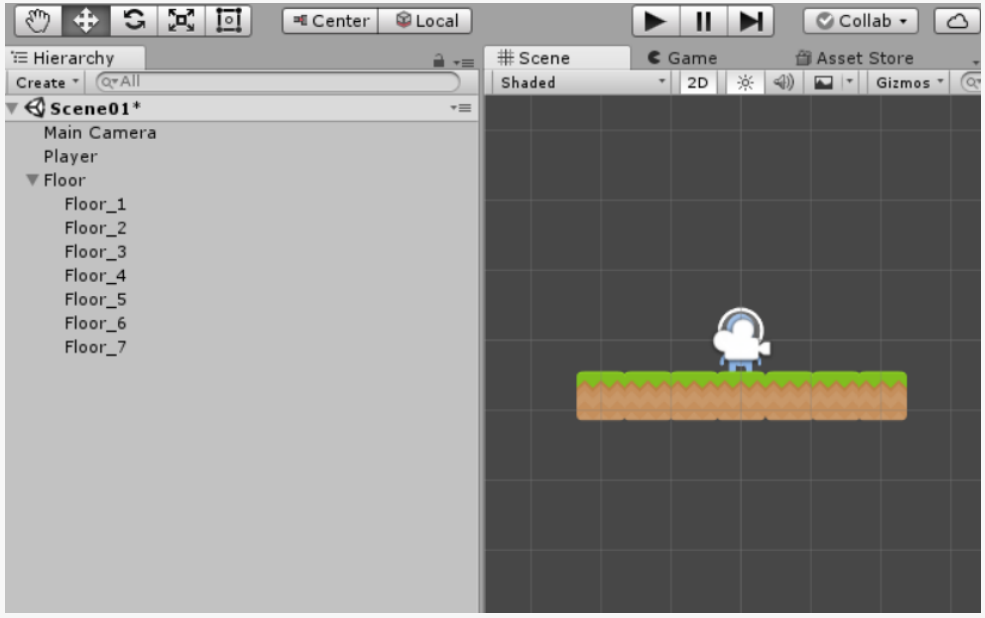
Move **titles_spritesheet_9** to the **hierarchy**. Move it to 0, (-0.8), 0



People
Innovation
Excellence

Visualize the platform

Duplicate 6 more tiles using CTRL+D and name it accordingly (The most left is Floor_1 and the far right is Floor_7

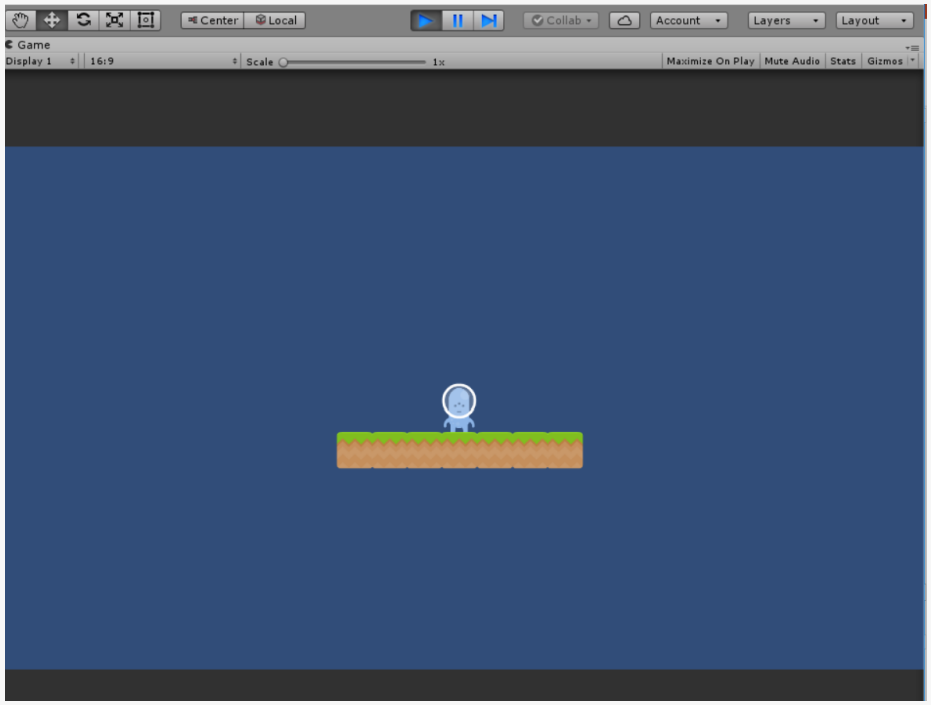


BINUS UNIVERSITY

People
Innovation
Excellence

Game Scene

Press the play button.. We'll see what happen with the screen



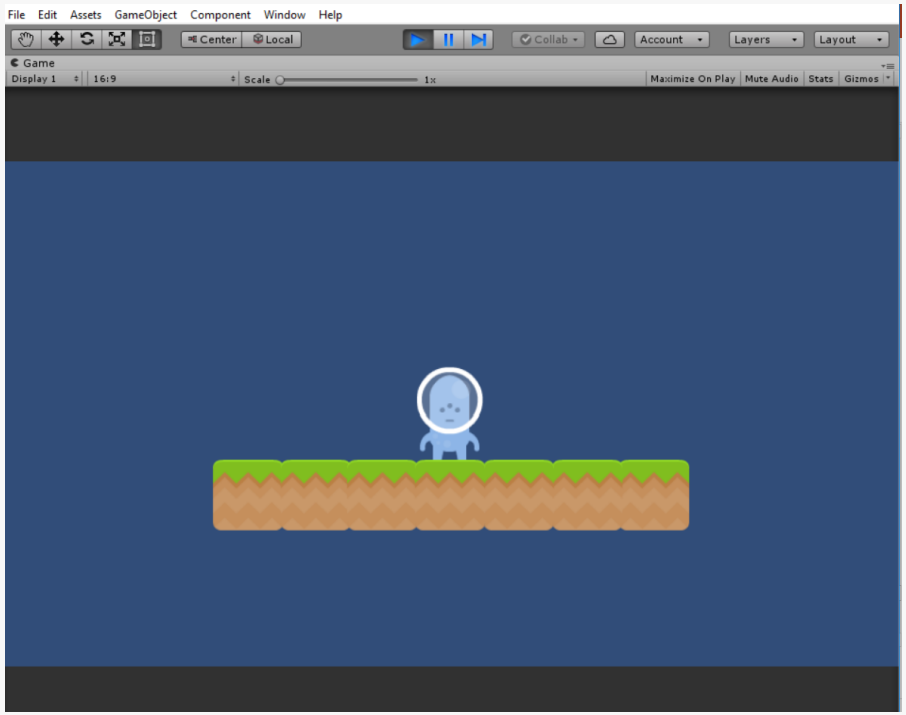
The character is too small!!!

BINUS UNIVERSITY

People
Innovation
Excellence

Game Scene

Double the size by scaling floor and player (2,2,1)*



Much better!!

**you may also play with your camera*

BINUS UNIVERSITY

Let's move....

Wait a minute.. Let's warm up with some script introduction..

People
Innovation
Excellence

BINUS UNIVERSITY

People
Innovation
Excellence

Why do we need to code?

In order to create any type of game, we need to understand code because the pieces of code are the instructions that tell the game engine (Unity) when and how you want something to occur at any given point.

BINUS UNIVERSITY

People
Innovation
Excellence

Scripting vs Programming

- Scripting languages are an alternative to programming languages, whereby we create lightweight, much more human-readable scripts that are interpreted by another program at runtime rather than compiled by the computer's processor directly as compiled languages are. Thus, scripting languages are normally slower than compiled languages.

BINUS UNIVERSITY

People
Innovation
Excellence

C# Data Structures

To create a variable, you simply

```
var name = “Dodick Sudirman”;
```

You can declare variable using the type that you preferred

```
string name2 = "Dozus";
int test = 15;
bool data = true;
float testFloat = 15.4f;
```

Notice that we use the f suffix for our float. Floats are numbers that contain decimals and are used for more granular positioning within the game world, as well as in the built-in physics engine

BINUS UNIVERSITY

People
Innovation
Excellence

C# Unity Code Structure

When you create Unity File, this is the content

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class HelloWorld : MonoBehaviour {
5      
6      // Use this for initialization
7      void Start () {
8
9      }
10
11     // Update is called once per frame
12     void Update () {
13
14     }
15 }
16
```

BINUS UNIVERSITY

People
Innovation
Excellence

Awake VS Start

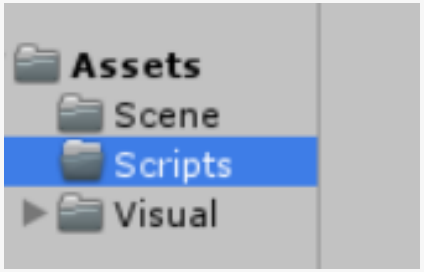
- Awake() is called only after a prefab is instantiated.
- If a GameObject is in-active during start up, Awake is not called until it is made active, or a function in any script attached to it is called.
- Awake is called first even if the script component is not enabled and is best used for setting up any resources between scripts and initialization.
- Start is called after Awake, immediately before the first Update, but only if the script component is enabled.
- This means that you can use Start for anything you need to occur when the script component is enabled. This allows you to delay any part of your initialization code until it's really needed.

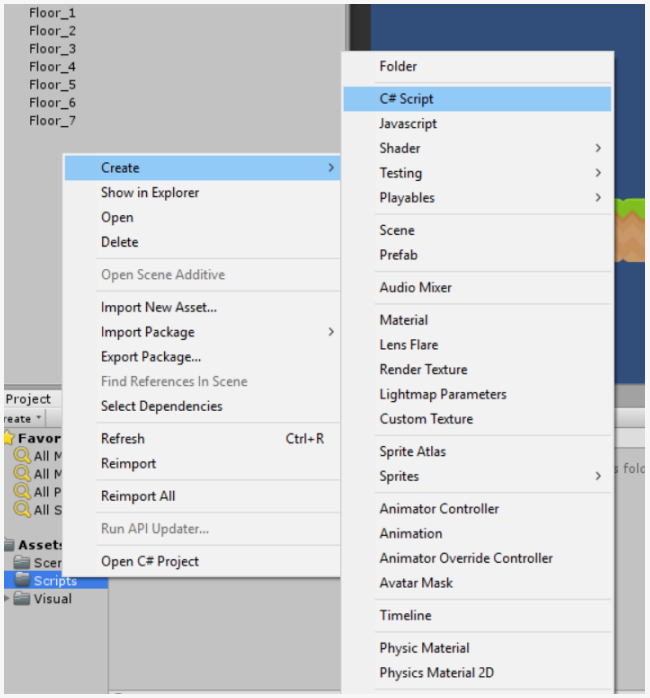
BINUS UNIVERSITY

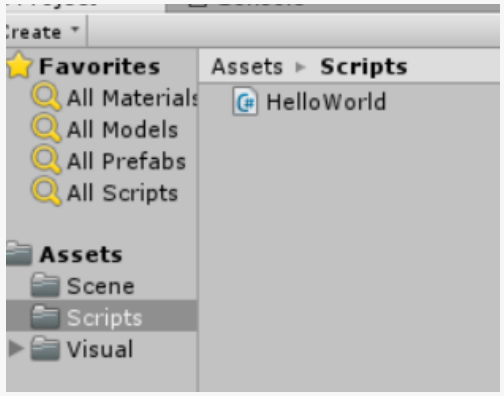
People
Innovation
Excellence

Let's start scripting

Create Scripts folder
Create a C# Script by right-clicking the Scripts folder and name it HelloWorld







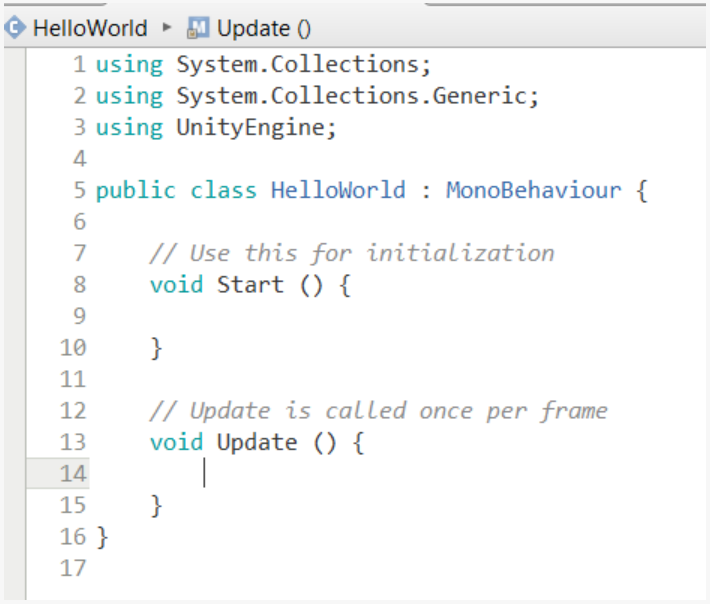
BINUS UNIVERSITY

People
Innovation
Excellence

The Script

Open the script by double clicking it...

You will see this...



BINUS UNIVERSITY

People
Innovation
Excellence

Let's do a "Hello World"

I know it's useless.. But let's get on with this...

HelloWorld.cs

selection

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class HelloWorld : MonoBehaviour {
6
7      // Use this for initialization
8      void Start () {
9          Debug.Log ("Hello World");
10     }
11
12     // Update is called once per frame
13     void Update () {
14
15     }
16 }
17
```

BINUS UNIVERSITY

People
Innovation
Excellence

What if you press play now?

NOTHING HAPPENED

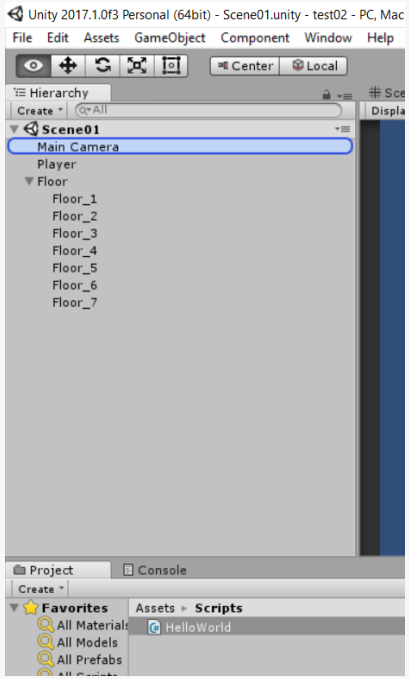
BINUS UNIVERSITY

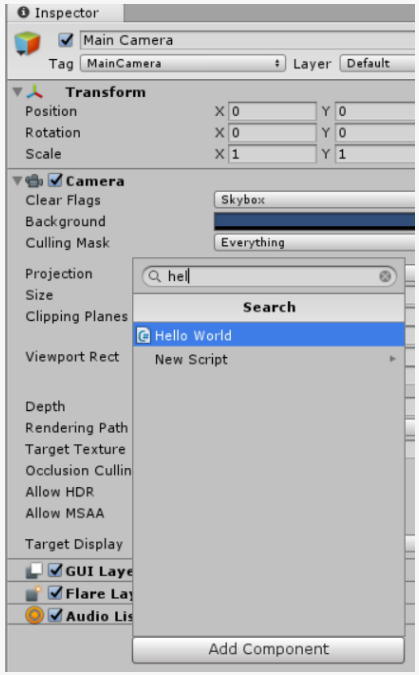
People
Innovation
Excellence

What should we do?

Attach the script to the game object

Since this is a useless script.. Just attached it to the main camera (you can use drag or drop or using add component function in the inspector)

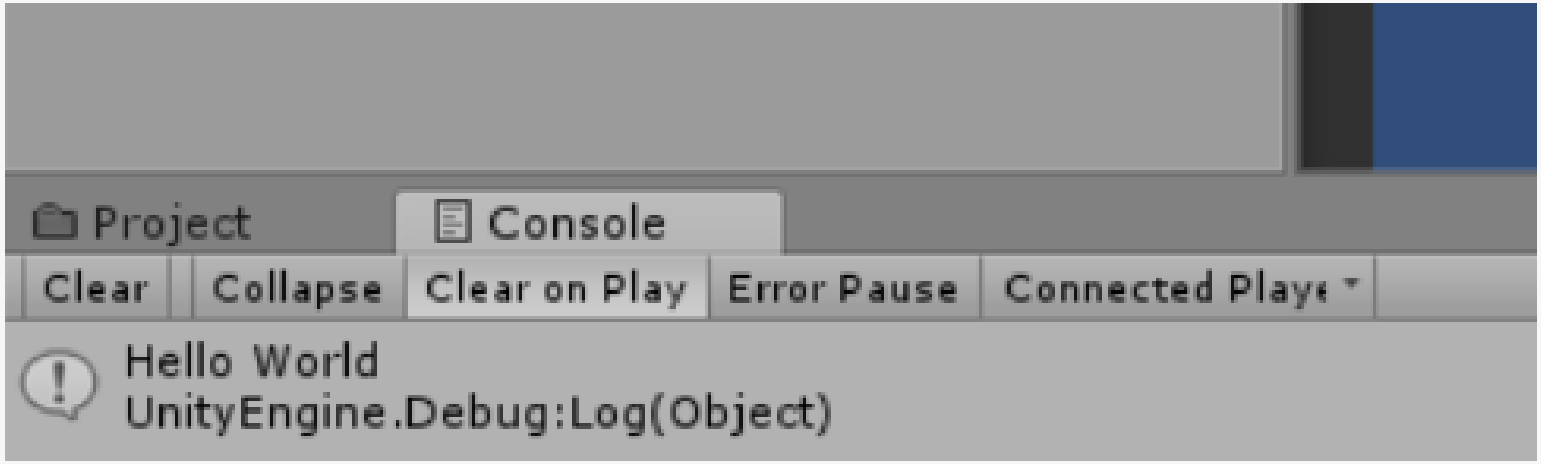




BINUS UNIVERSITY

People
Innovation
Excellence

Run the game now!!



YEAY!!!!!!!!!!!!!!!!!!

BINUS UNIVERSITY

Let's make something different

Modify the script and save it

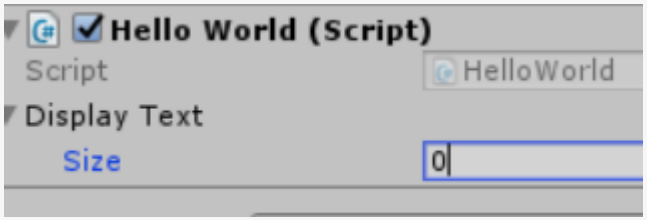
People
Innovation
Excellence

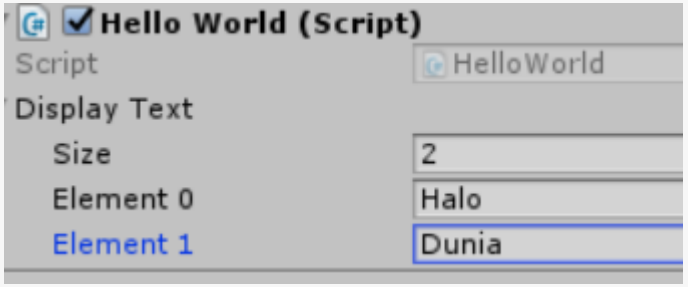
BINUS UNIVERSITY

People
Innovation
Excellence

Let's make something different

Open the game inspector and modify the parameter



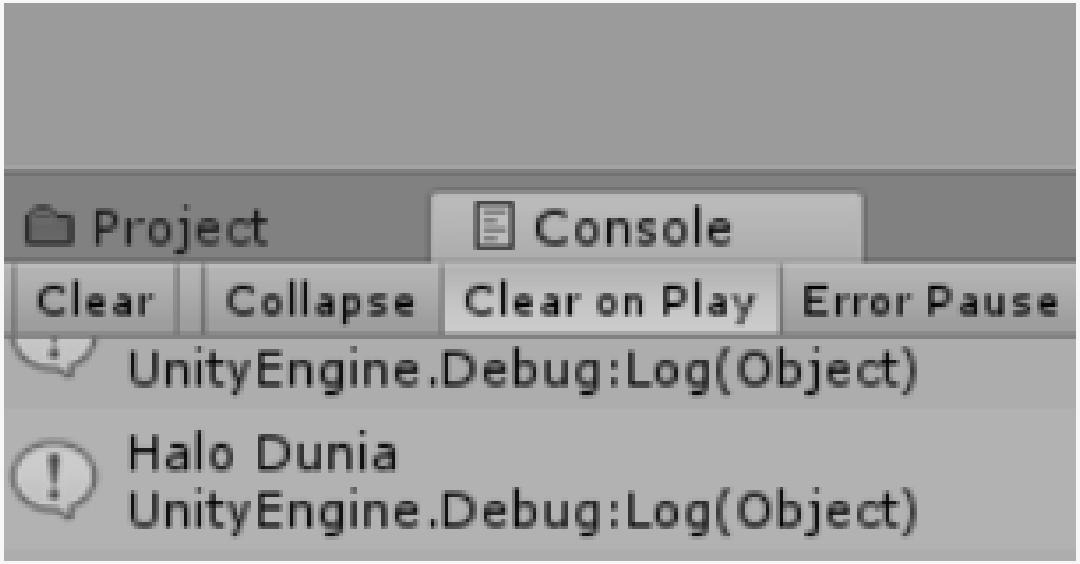


BINUS UNIVERSITY

People
Innovation
Excellence

Let's make something different

RUN IT!!!

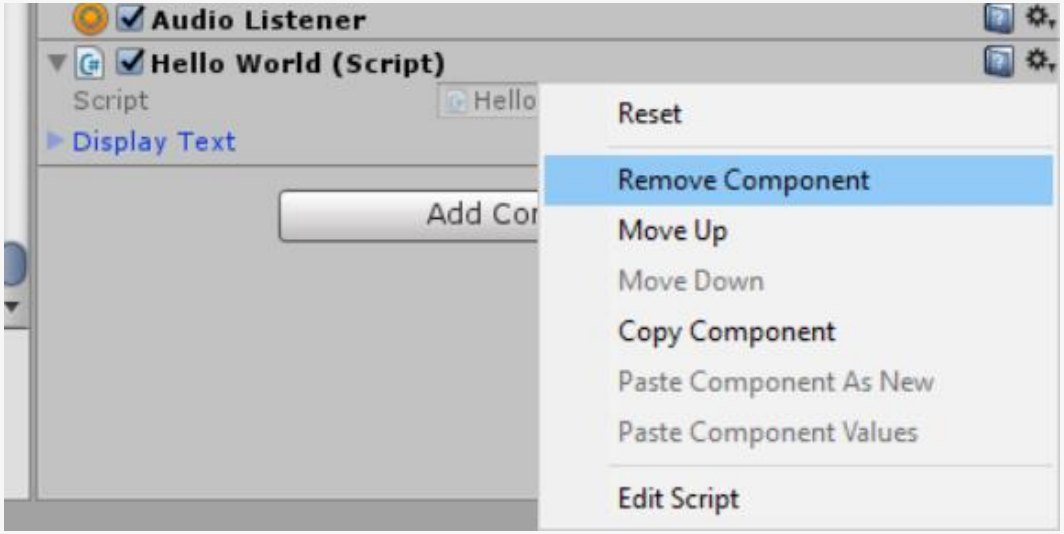


BINUS UNIVERSITY

People
Innovation
Excellence

Let's make something usefull

Remove the script from the main camera...

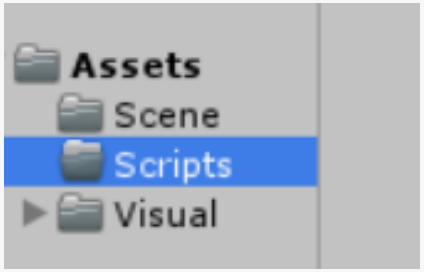


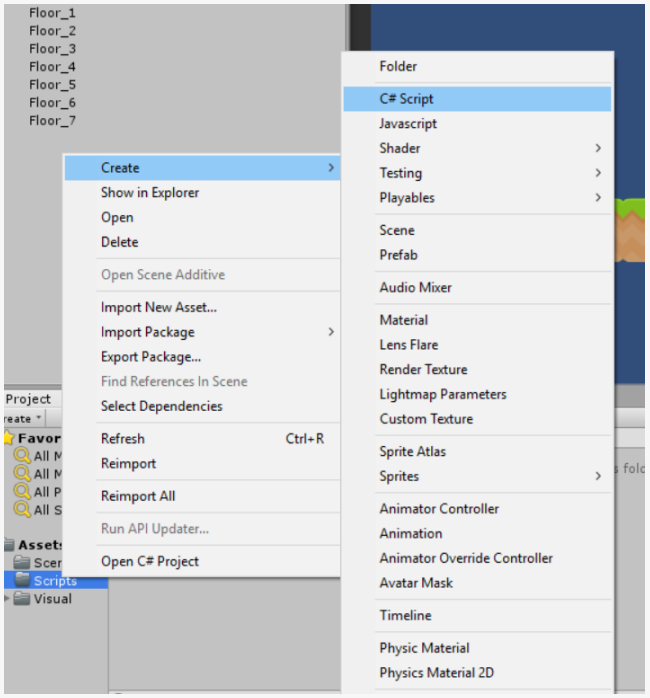
BINUS UNIVERSITY

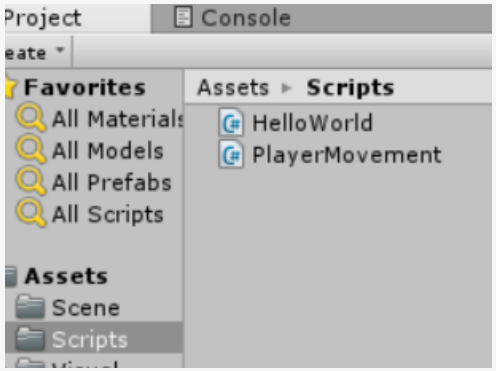
People
Innovation
Excellence

Let's start scripting something useful

Create a C# Script by right-clicking the Scripts folder and name it PlayerMovement







BINUS UNIVERSITY

People
Innovation
Excellence

Let's add some variables

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class PlayerMovement : MonoBehaviour {
6
7
8      public float speed = 10.0f;
9      private float leftWall = -4f;
10     private float rightWall = 4f;
11
12     // Use this for initialization
13     void Start () {
14
15     }
16
17     // Update is called once per frame
18     void Update () {
19
20     }
21 }
22
```

Speed variable indicates the player speed movement

leftWall and rightWall variables used to limit player movement area

BINUS UNIVERSITY

People
Innovation
Excellence

Let's add some logic

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerMovement : MonoBehaviour {


    public float speed = 10.0f;
    private float leftWall = -4f;
    private float rightWall = 4f;

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {
        float translation = Input.GetAxis("Horizontal") * speed * Time.deltaTime;
        if (transform.position.x + translation < rightWall &&
            transform.position.x + translation > leftWall)
            transform.Translate(translation, 0, 0);

    }
}
```

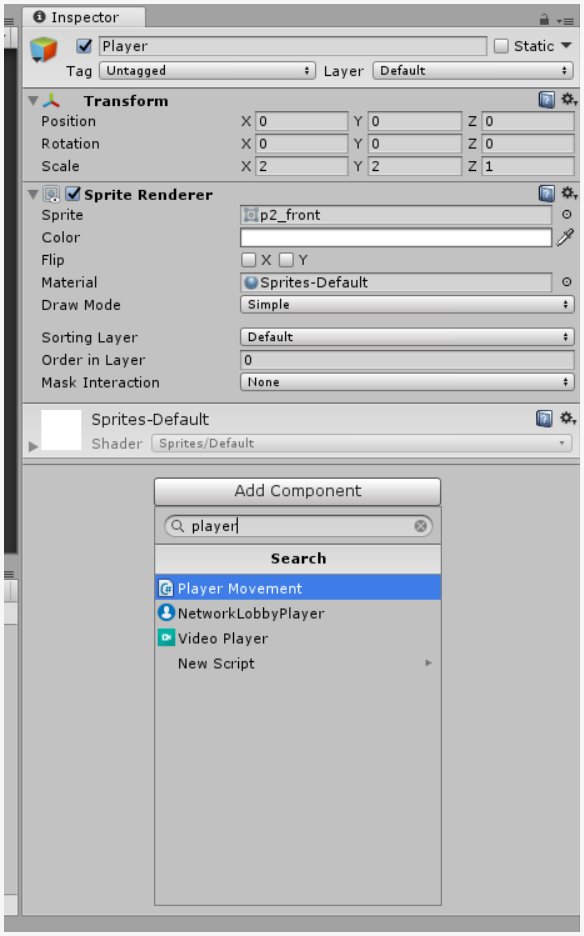
in the Update() function we first need to calculate how far it will be translated and where, based on the input of the player. In this case, we take the horizontal axis, which, by default, is bound to the arrow keys. In order to calculate this, we need to take into consideration how much time is passed from the last frame. This can be done by adding Time.deltaTime into the equation. Then, we need to translate the character so it doesn't fall off the boundary after the translation.

BINUS UNIVERSITY

People
Innovation
Excellence

Activate the script

Add the script to the Player Object

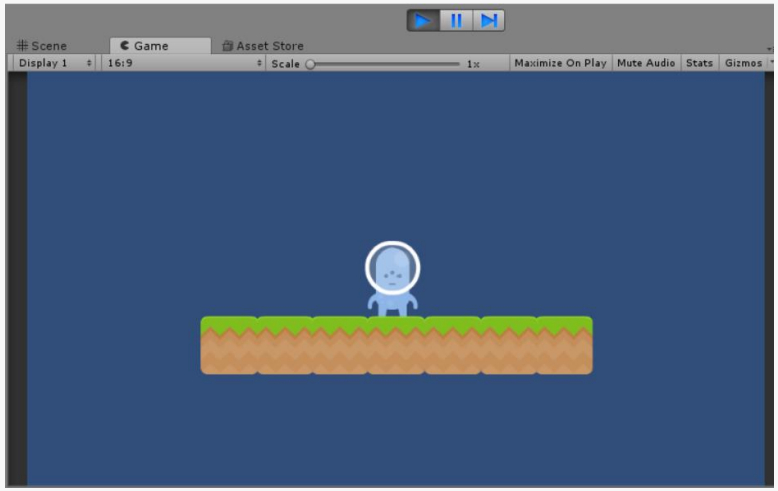


BINUS UNIVERSITY

People
Innovation
Excellence

It's alive!!!!!!!!!!!!

RUN IT!!!!!!!!!!!!!!

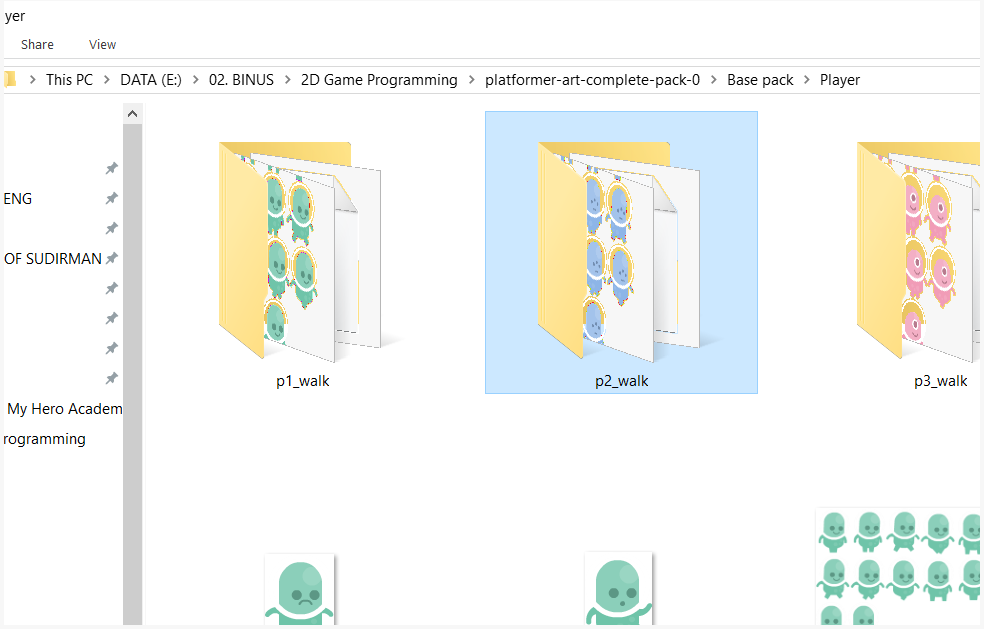


BINUS UNIVERSITY

People
Innovation
Excellence

Create some animation...

Search for p2_walk folder

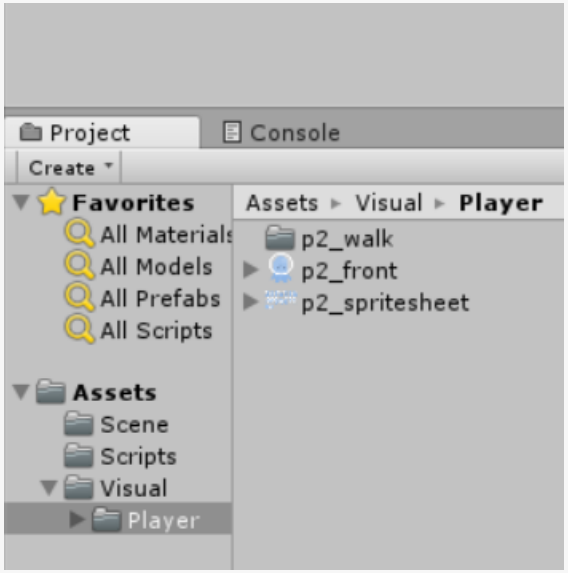


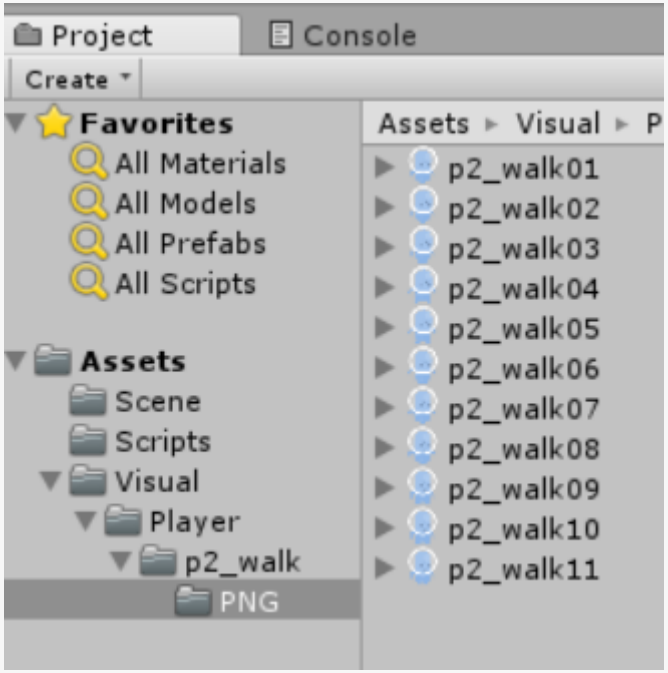
BINUS UNIVERSITY

People
Innovation
Excellence

Create some animation...

Move it to the project's player folder and check out the PNG folder



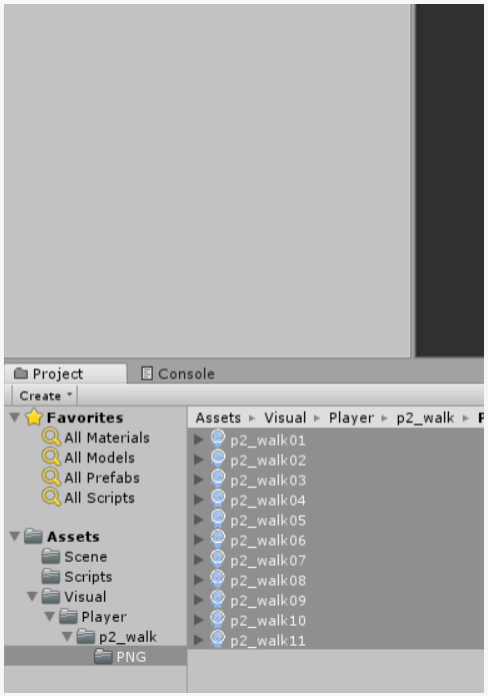


BINUS UNIVERSITY

People
Innovation
Excellence

Create some animation...

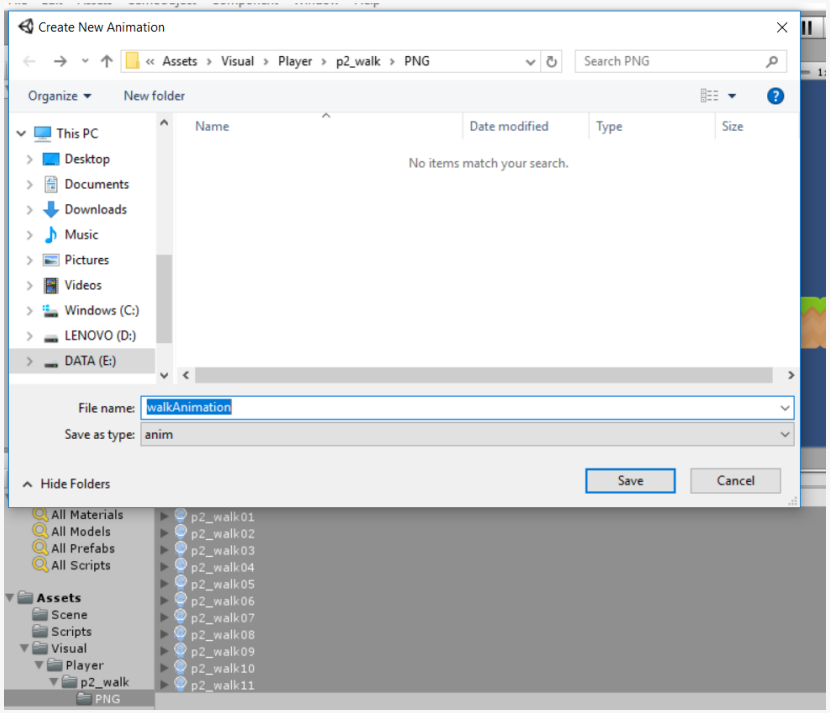
Highlight all and move it to the



BINUS UNIVERSITY

People
Innovation
Excellence

Create some animation...

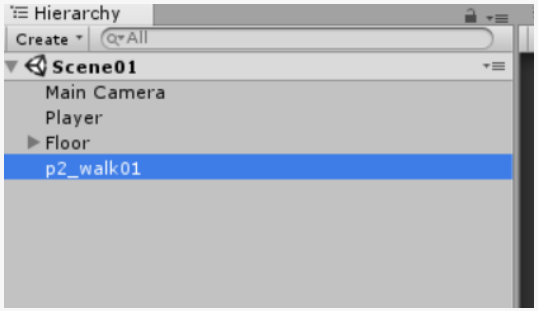


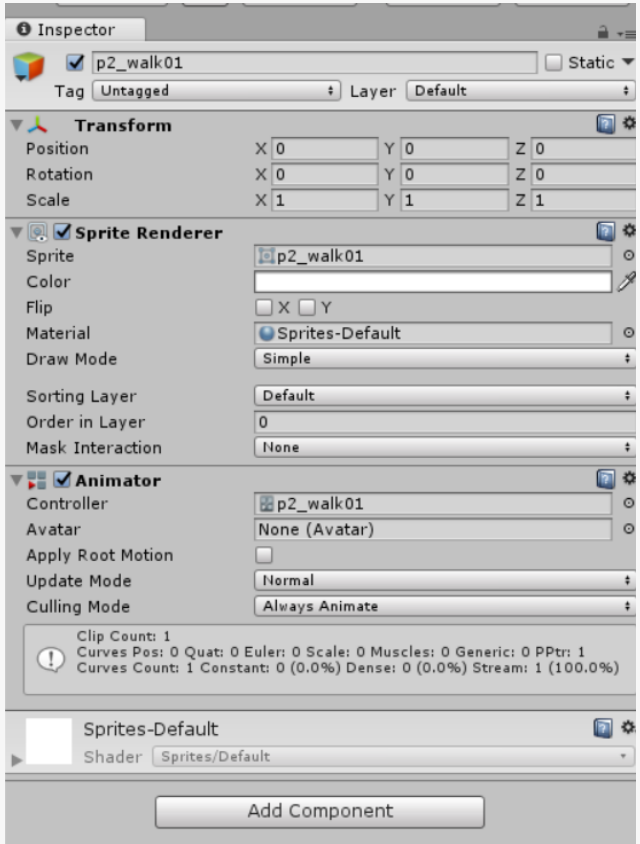
BINUS UNIVERSITY

People
Innovation
Excellence

Create some animation...

Highlight the new object and observe the inspector



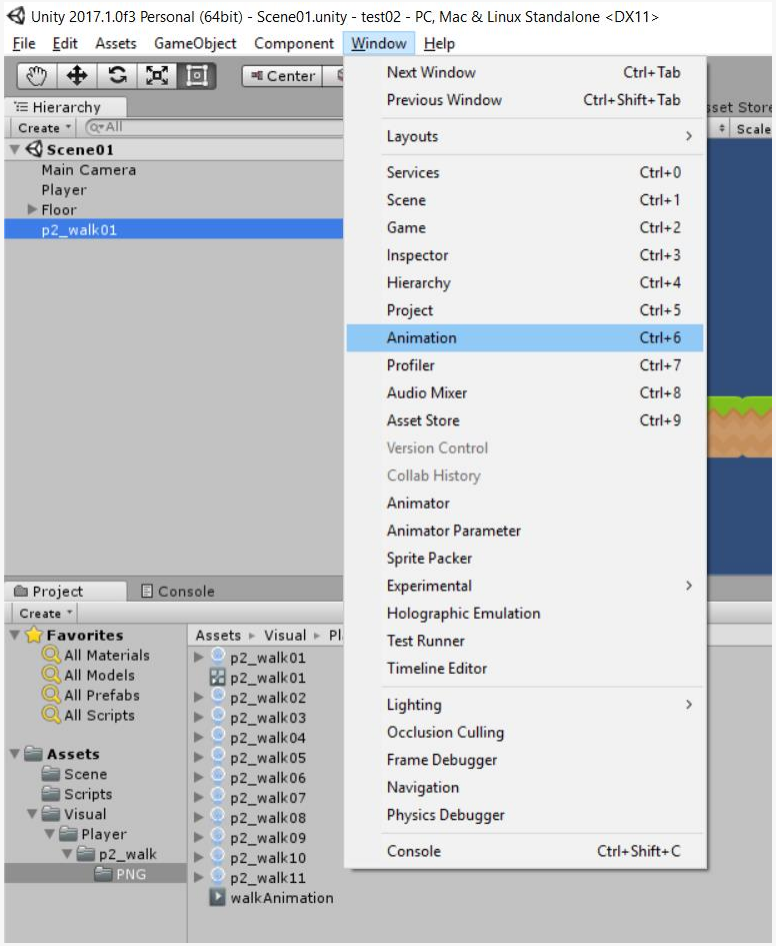


BINUS UNIVERSITY

People
Innovation
Excellence

Another way to create new animation

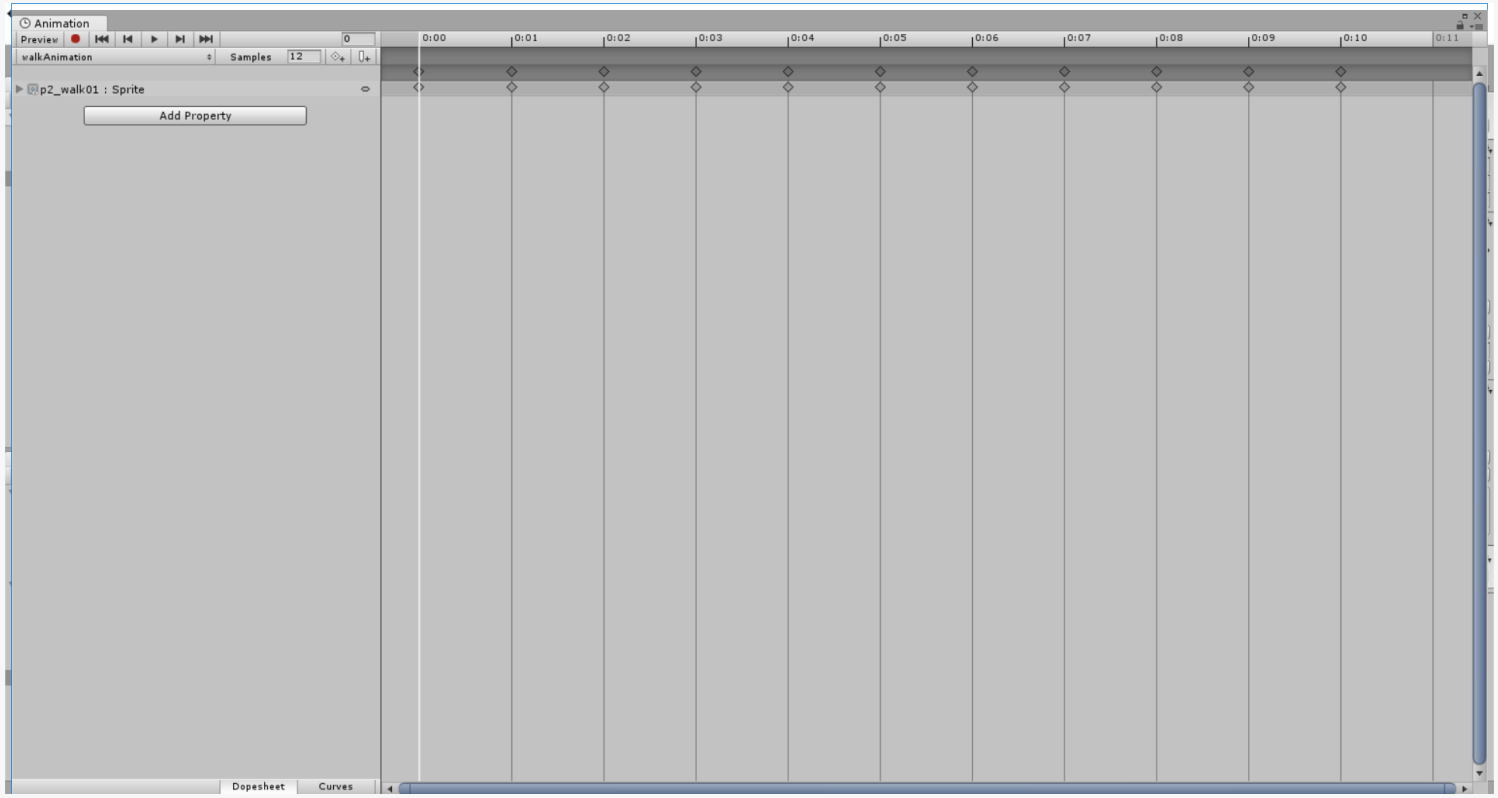
Open window toolbar and click the animation option



BINUS UNIVERSITY

People
Innovation
Excellence

Animation Pane

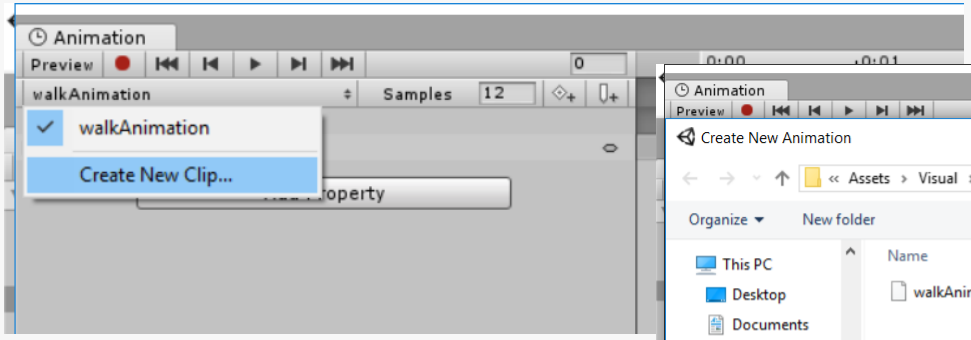


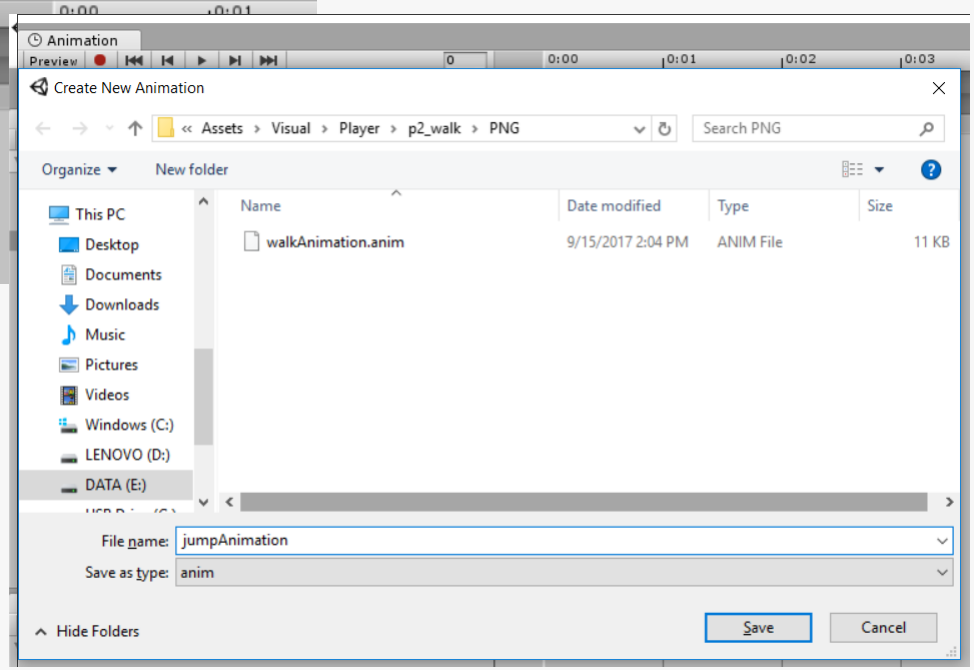
BINUS UNIVERSITY

People
Innovation
Excellence

Create new clip...

Click the walkAnimation and proceed to click Create New Clip...
Name it jumpAnimation.anim



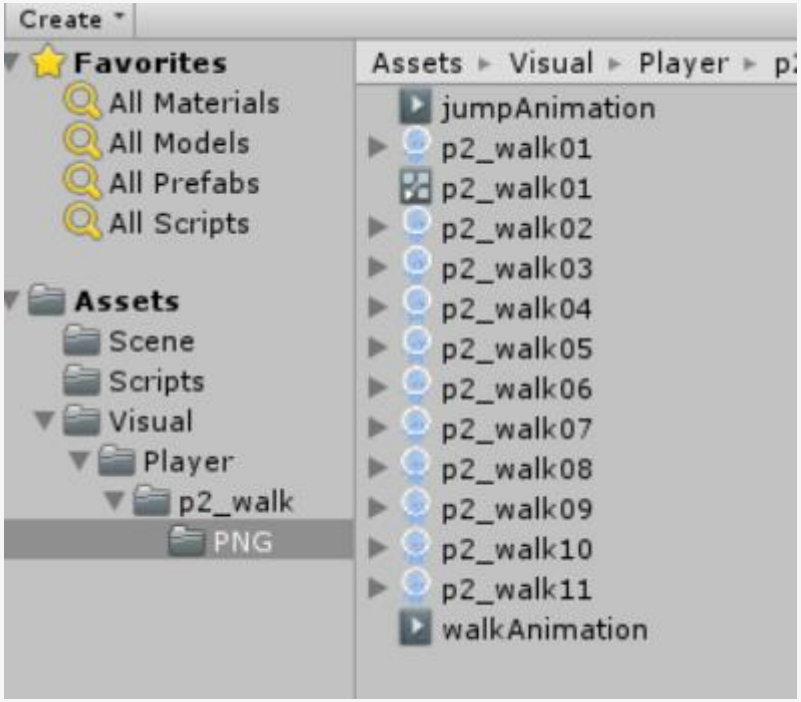


BINUS UNIVERSITY

People
Innovation
Excellence

Your animation clip...

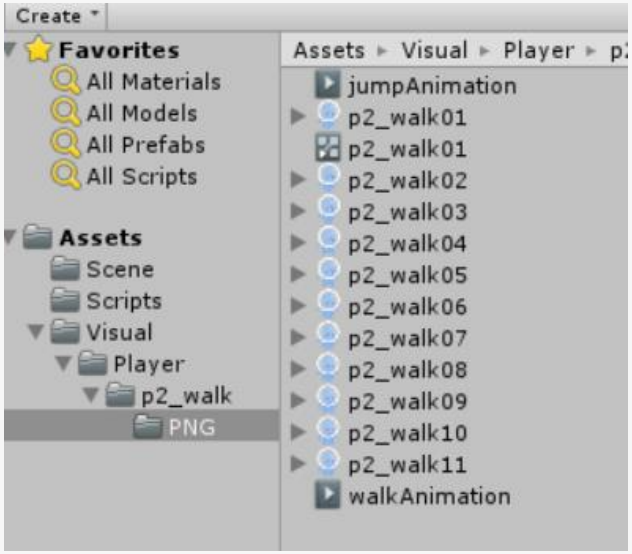
Your files are available where you save it.



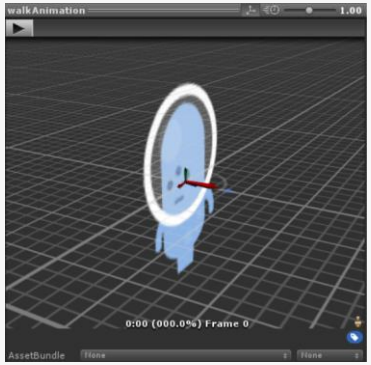
BINUS UNIVERSITY

People
Innovation
Excellence

Animate it



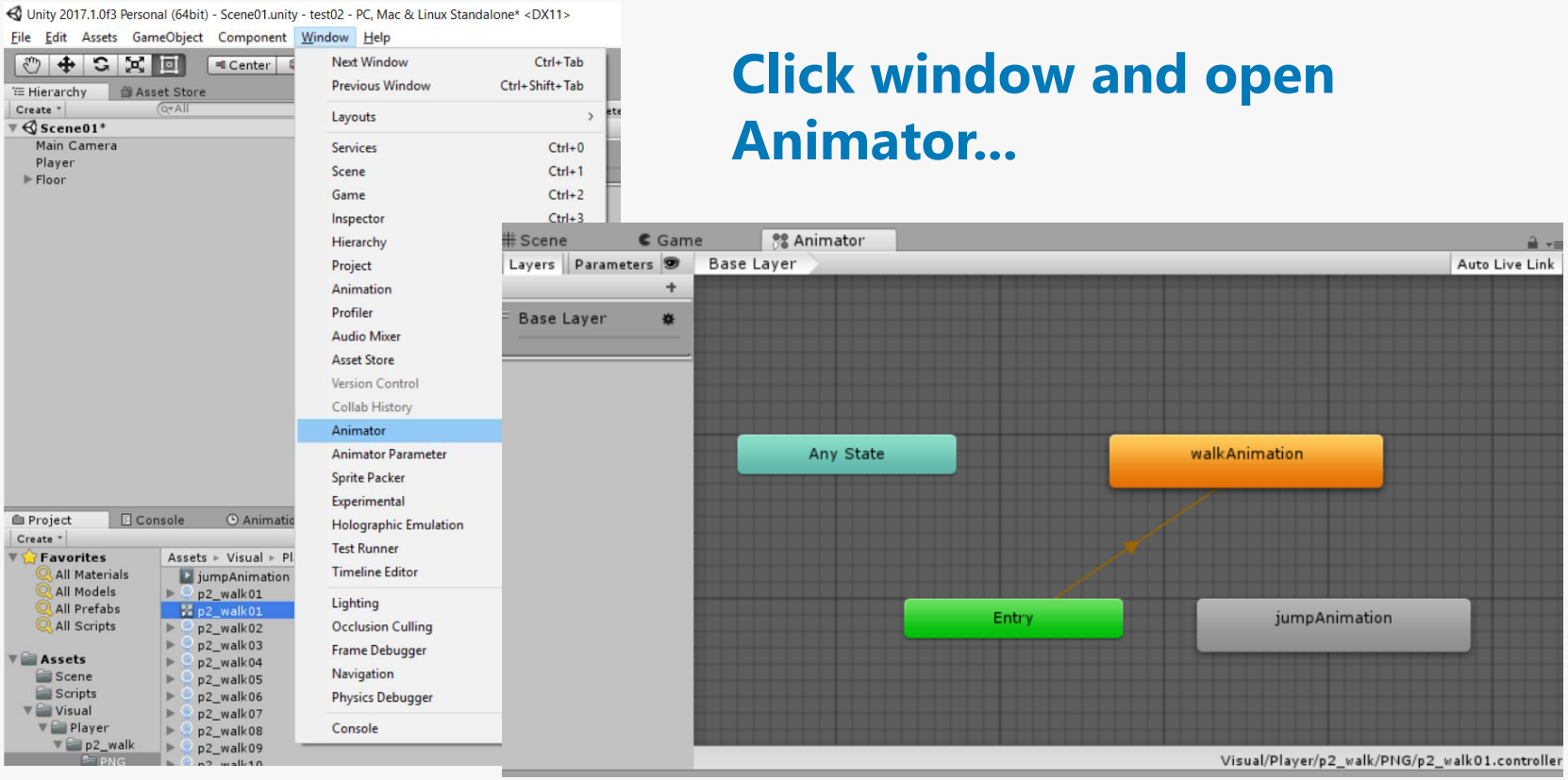
Select the walk animation and open the inspector. Drag the player object to the preview windows.



BINUS UNIVERSITY

People
Innovation
Excellence

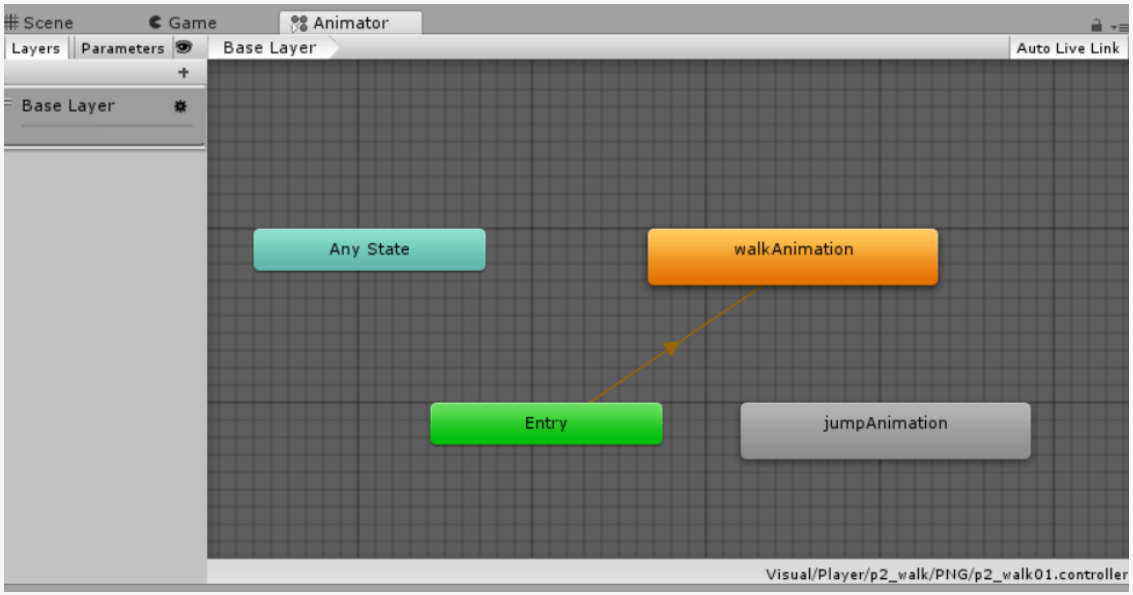
Handling Animation



Click window and open Animator...

BINUS UNIVERSITY

People
Innovation
Excellence

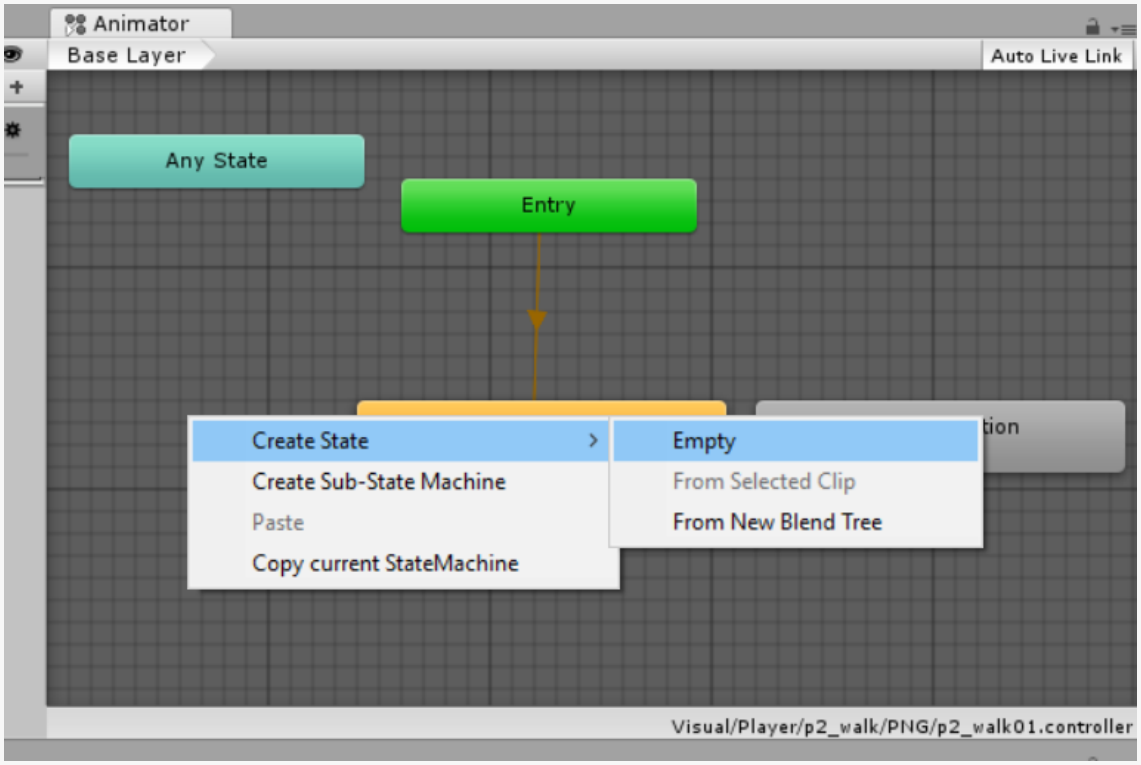


- **Entry node** (marked green): This is used when transitioning into a state machine, provided the required conditions were met.
- **Exit node** (marked red): This is used to exit a state machine when the conditions have been changed or completed. By default it is not present, as there isn't one in the previous image.
- **Default node** (marked orange): This is the default state of the **Animator** and is automatically transitioned to from the entry node.
- **Sub-state nodes** (marked grey): These are also called custom nodes. They are used typically to represent a state for an object where an event will occur (in our case, an animation will be played).
- **Transitions** (arrows): These allow state machines to switch between one another by setting the conditions that will be used by Animator to decide which state will be activated.

BINUS UNIVERSITY

People
Innovation
Excellence

New state animation

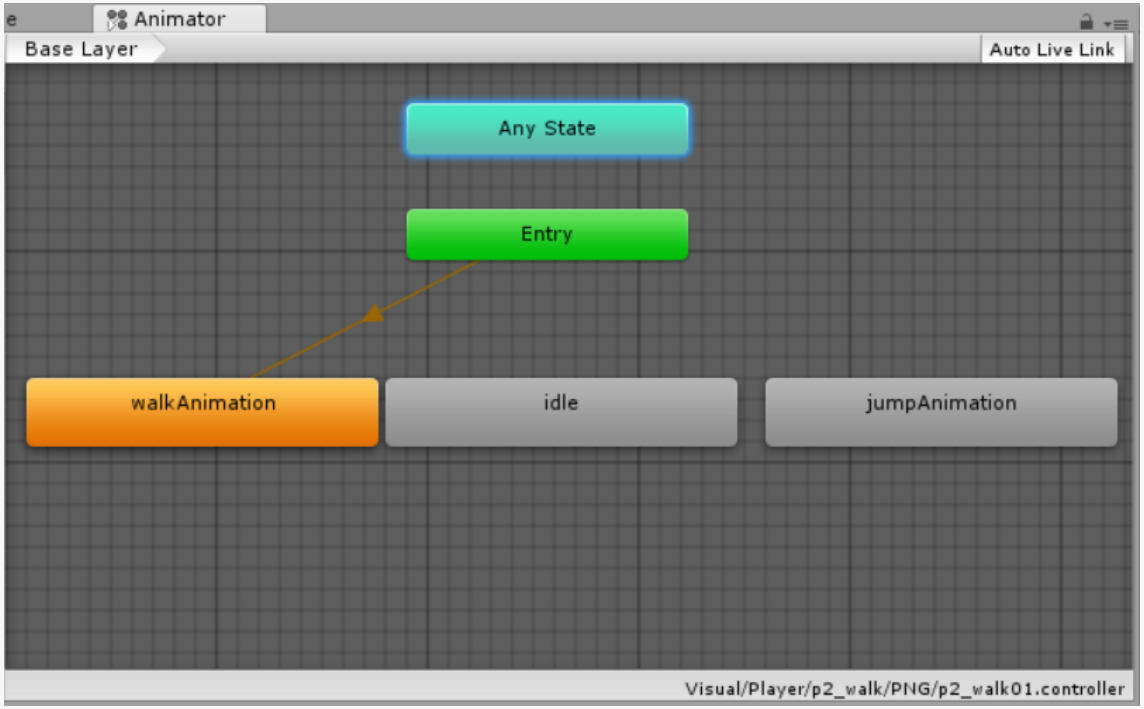


Let's create a new state called idle

BINUS UNIVERSITY

People
Innovation
Excellence

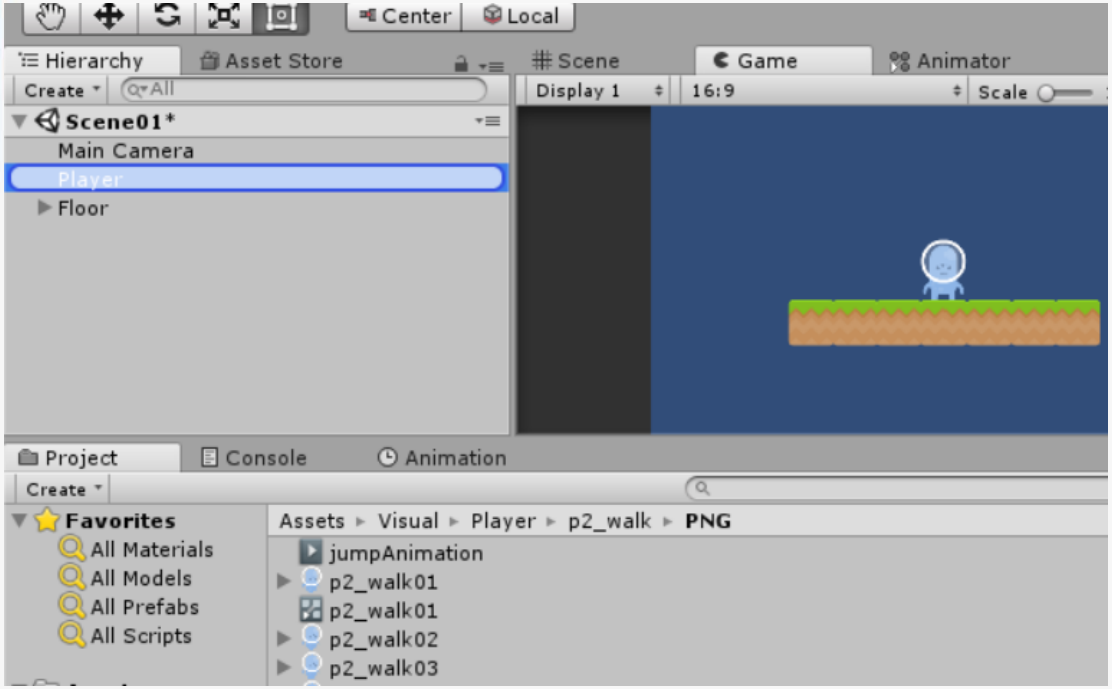
Clean up the animation state



BINUS UNIVERSITY

People
Innovation
Excellence

Applying the animation

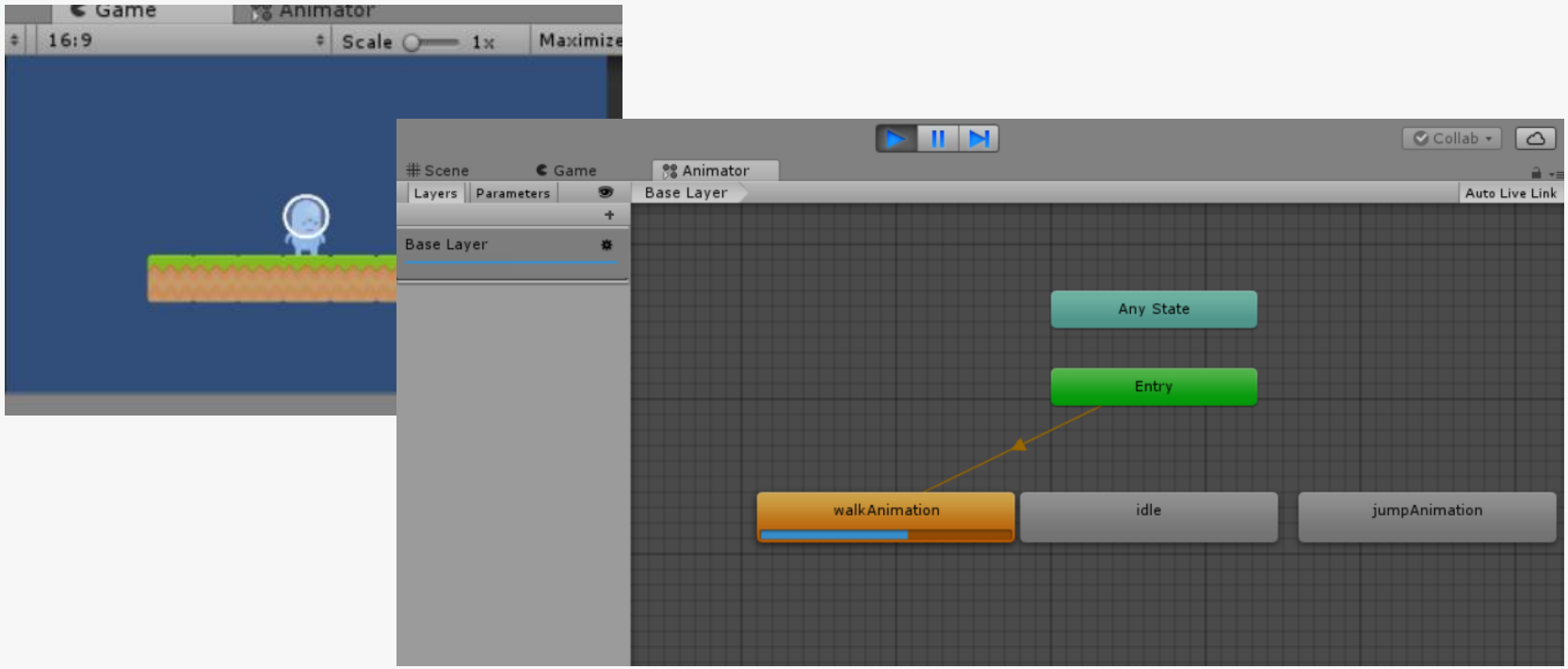


Drag p2_walk01 to the Player object

BINUS UNIVERSITY

People
Innovation
Excellence

It's walking!



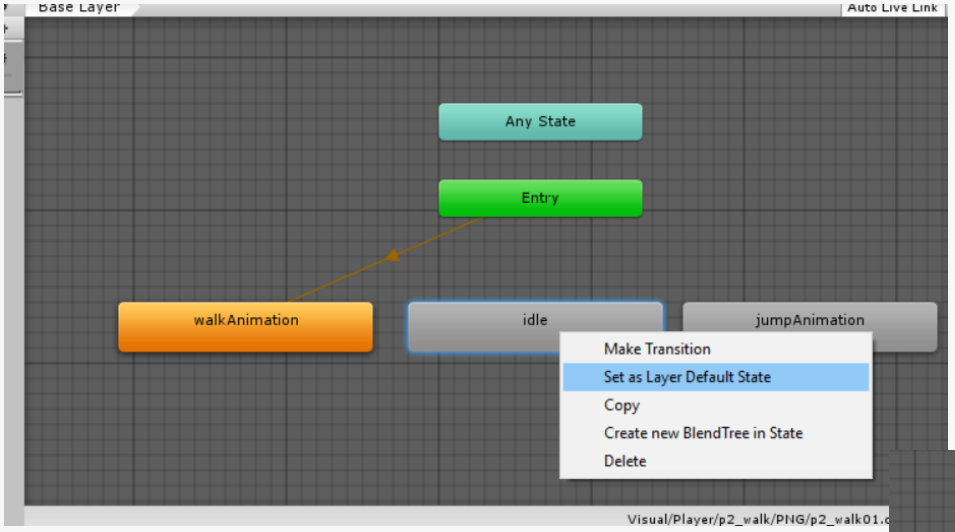
Check the animator pane while the game is running

BINUS UNIVERSITY

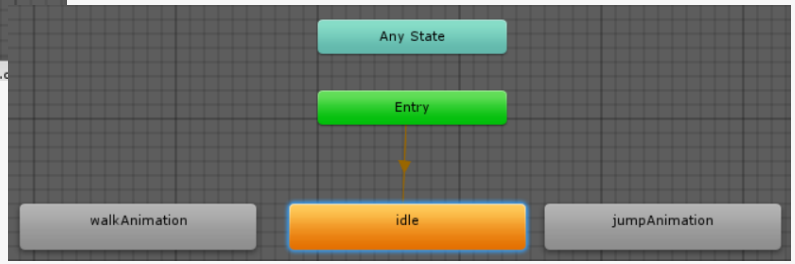
People
Innovation
Excellence

Make more interactive

Map it to the idle state to start with no animation



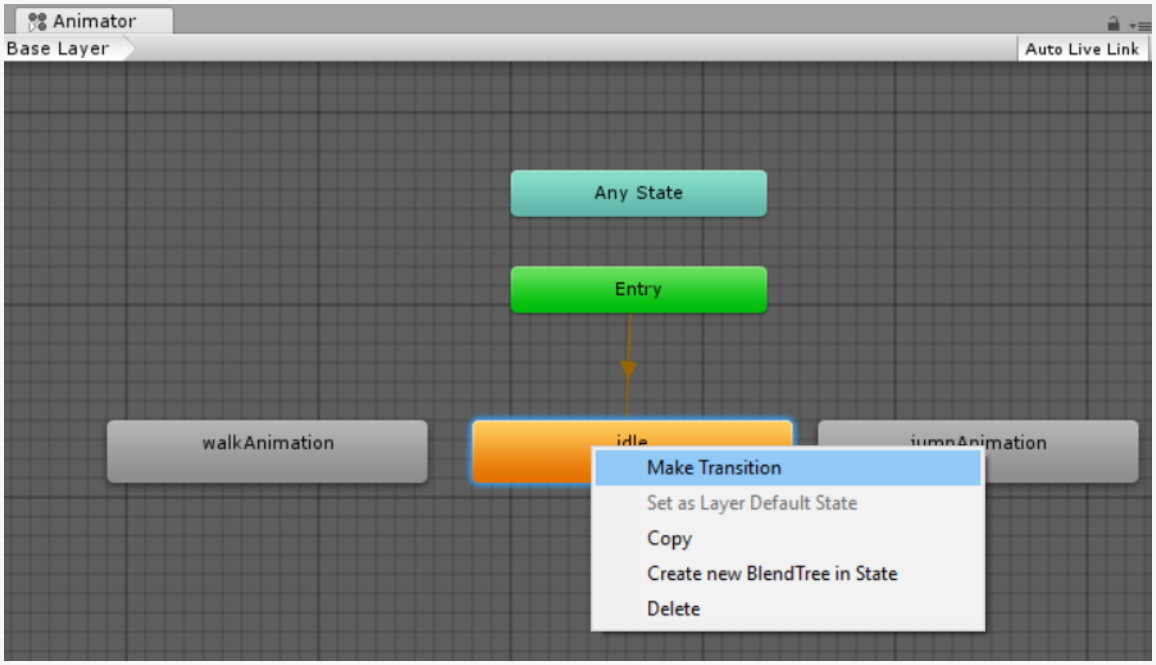
Open the animator, right click the idle state and pick “Set as Layer Default State”



BINUS UNIVERSITY

People
Innovation
Excellence

Let's connect a transition

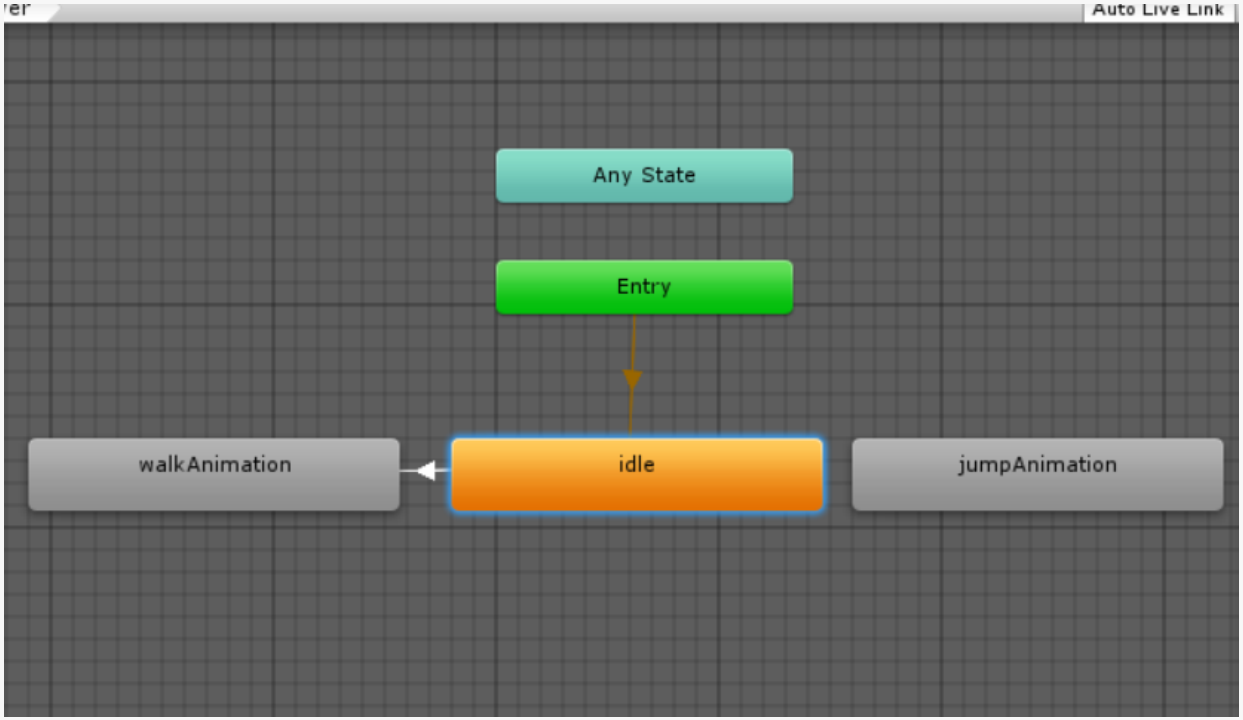


Right click the idle state and pick the "Make transition" option.

BINUS UNIVERSITY

People
Innovation
Excellence

Let's connect a transition

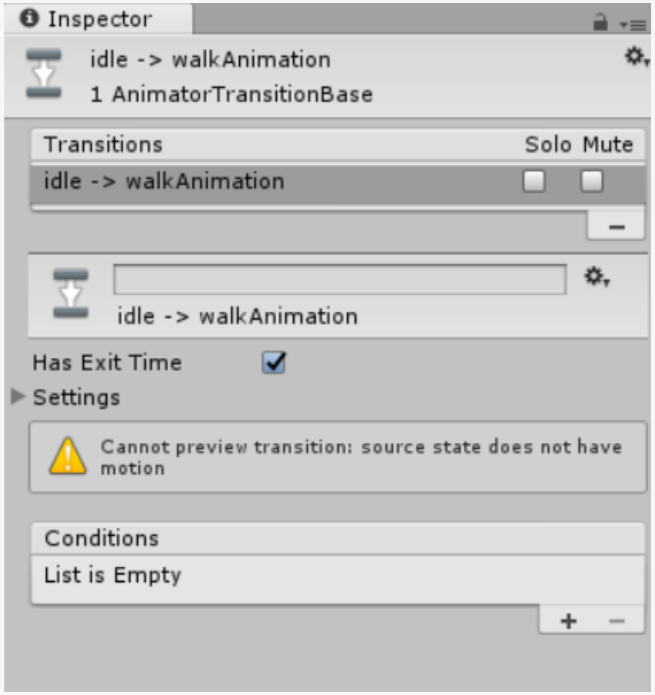


Connect it with the walkAnimation state

BINUS UNIVERSITY

People
Innovation
Excellence

The inspector



Click the arrow and open the inspector.
More information:

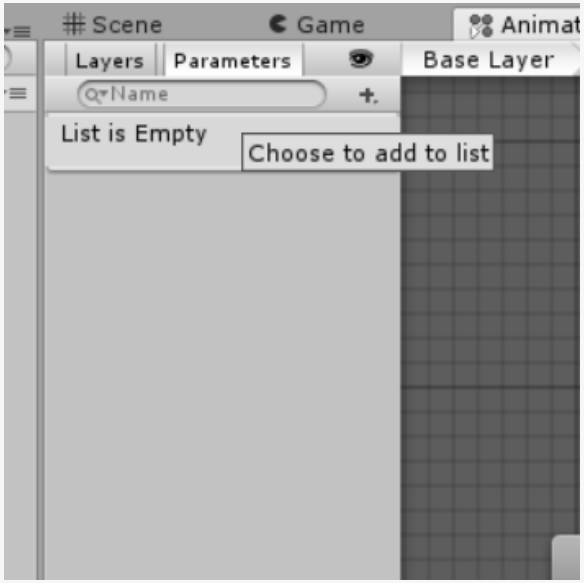
- **Name** (optional): We can assign a name to the transition. This is useful to keep everything organized and easy to access. In this case, let's name this transition StartWalking.
- **Has Exit Time**: Whether or not the animation should be played to the end before exiting its state when the conditions are not being met any more.
- **Conditions**: The conditions that should be met so that the transition takes place.

BINUS UNIVERSITY

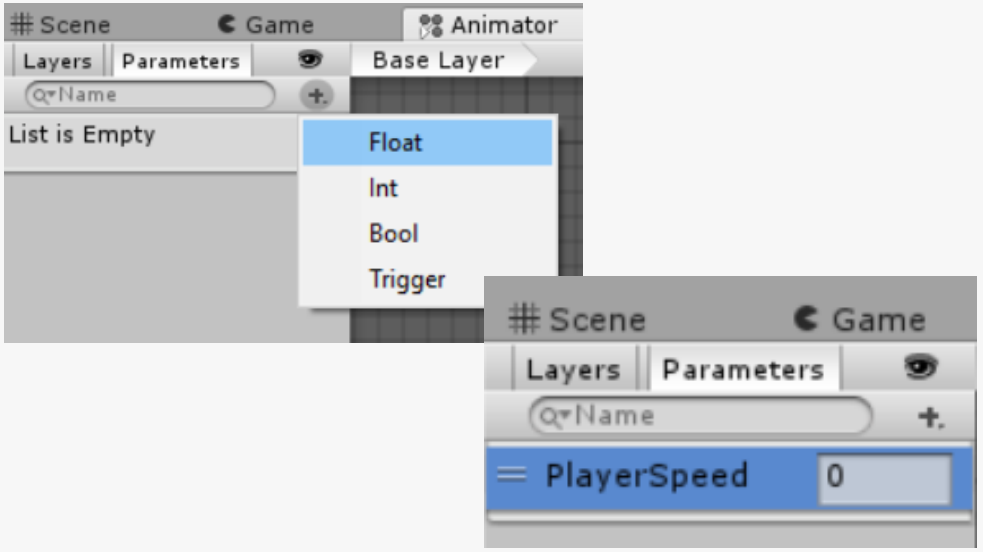
People
Innovation
Excellence

Let's create a parameter

The parameter will be used as a variable of your animation



Click the plus button to add new animation

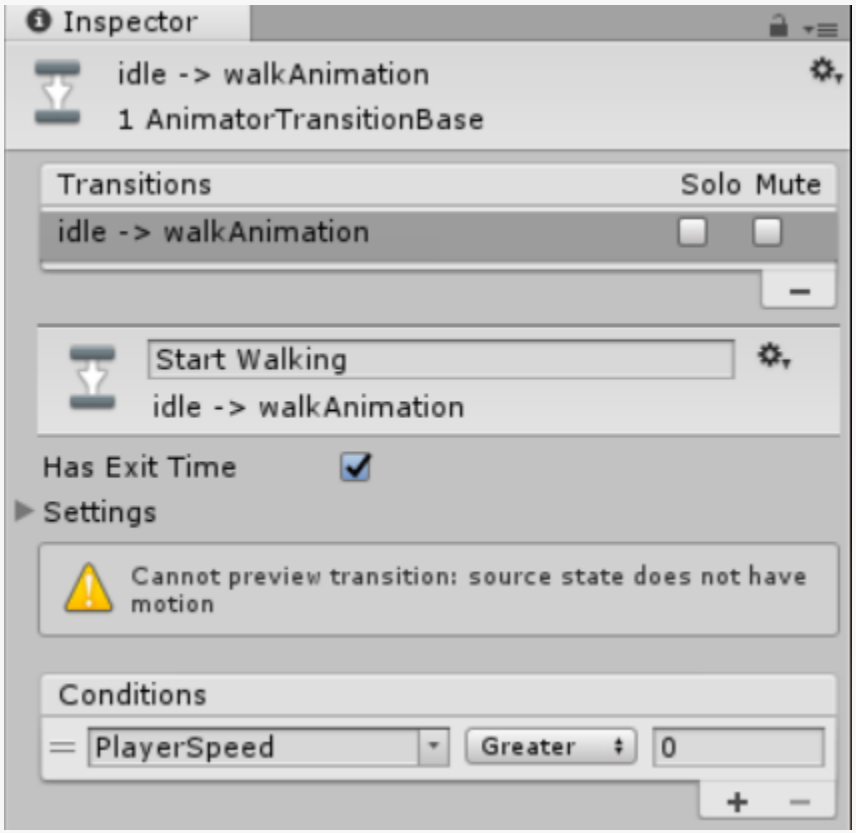


Pick float for the new variable and name it PlayerSpeed

BINUS UNIVERSITY

People
Innovation
Excellence

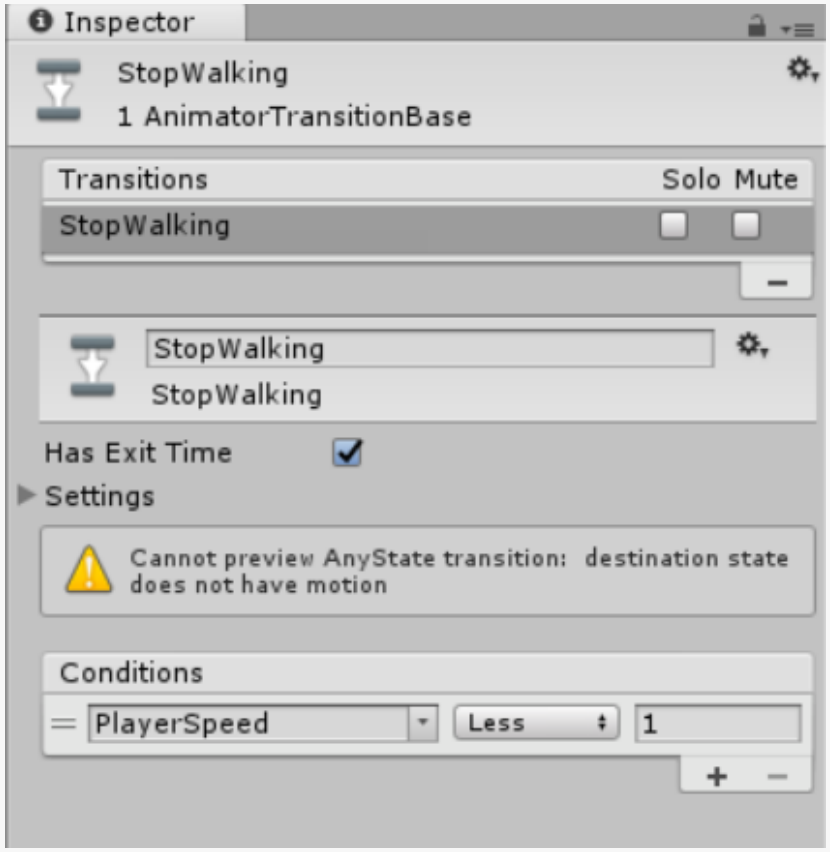
Edit the inspector



Name the transition Start Walking and add new parameter on the Conditions where it is Greater than 0.

BINUS UNIVERSITY

People
Innovation
Excellence



CREATE A REVERSE TRANSITION CALLED STOP WALKING!

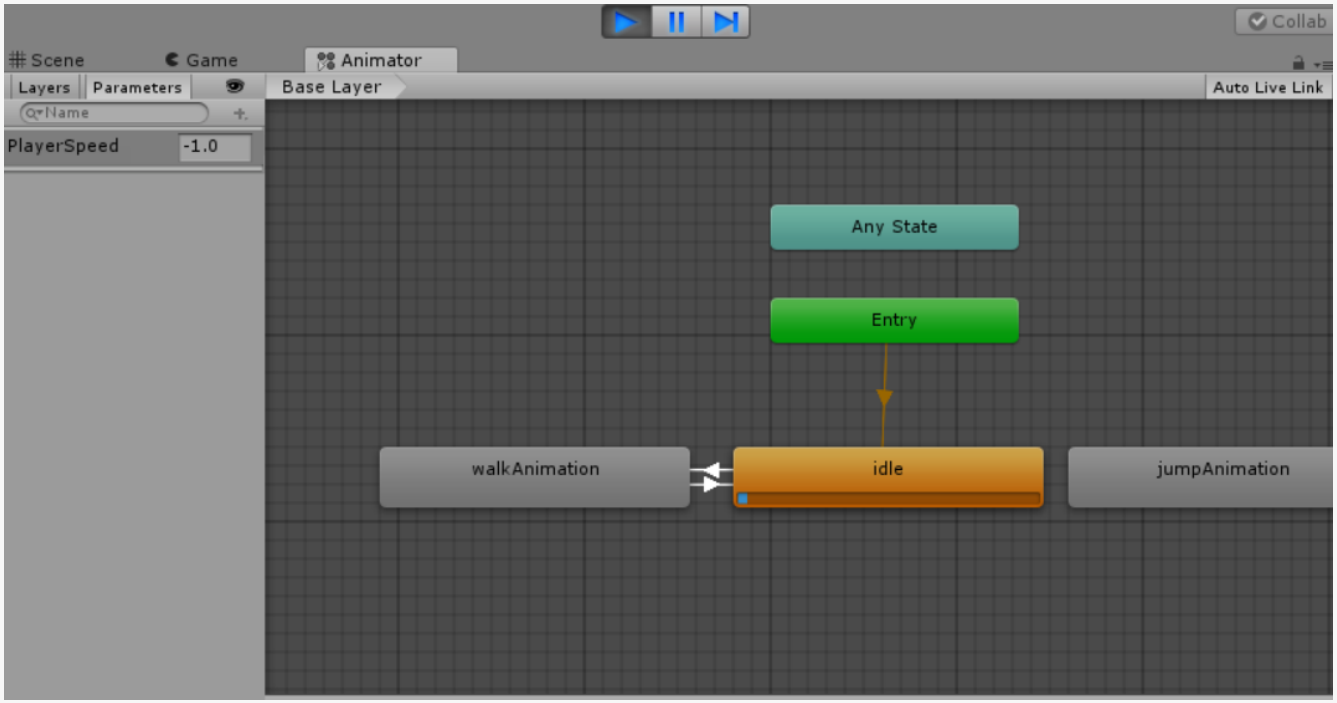
This transition will use the parameter PlayerSpeed so that when its value becomes less than one the transition is triggered.

BINUS UNIVERSITY

People
Innovation
Excellence

TRY IT OUT...

Run the game and change the parameter



BINUS UNIVERSITY

People
Innovation
Excellence

Continue to the script

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class PlayerMovement : MonoBehaviour {
6
7
8     public float speed = 10.0f;
9     private float leftWall = -4f;
10    private float rightWall = 4f;
11    private Animator anim;
```

Create animator variable and name it Anim (check line 11)

BINUS UNIVERSITY

People
Innovation
Excellence

Get the reference

```
5  public class PlayerMovement : MonoBehaviour {
6
7
8      public float speed = 10.0f;
9      private float leftWall = -4f;
10     private float rightWall = 4f;
11     private Animator anim;
12
13     // Use this for initialization
14     void Start () {
15         anim = GetComponent<Animator>();
16     }
```

In the Start() function, we need to get the reference to the Animator by calling the GetComponent() function (check line 15).

BINUS UNIVERSITY

```
19      void Update () {
20          // Get the horizontal axis that by default is bound to the arrowkeys
21          // The value is in the range -1 to 1
22          // Make it move per seconds instead of frames
23          float translation = Input.GetAxis("Horizontal") * speed * Time.deltaTime;
24          //Change direction if needed
25          if (translation > 0) {
26              transform.localScale = new Vector3(2, 2, 2);
27          }
28          else if (translation < 0) {
29              transform.localScale = new Vector3(-2, 2, 2);
30          }
```

People
Innovation
Excellence

We need to change the Update() function so that we can take care of the Animator and change its parameters accordingly. As a result, our character will be animated. Let's start to add this piece of code between the two lines of code we had before, just before the if statement

BINUS UNIVERSITY

People
Innovation
Excellence

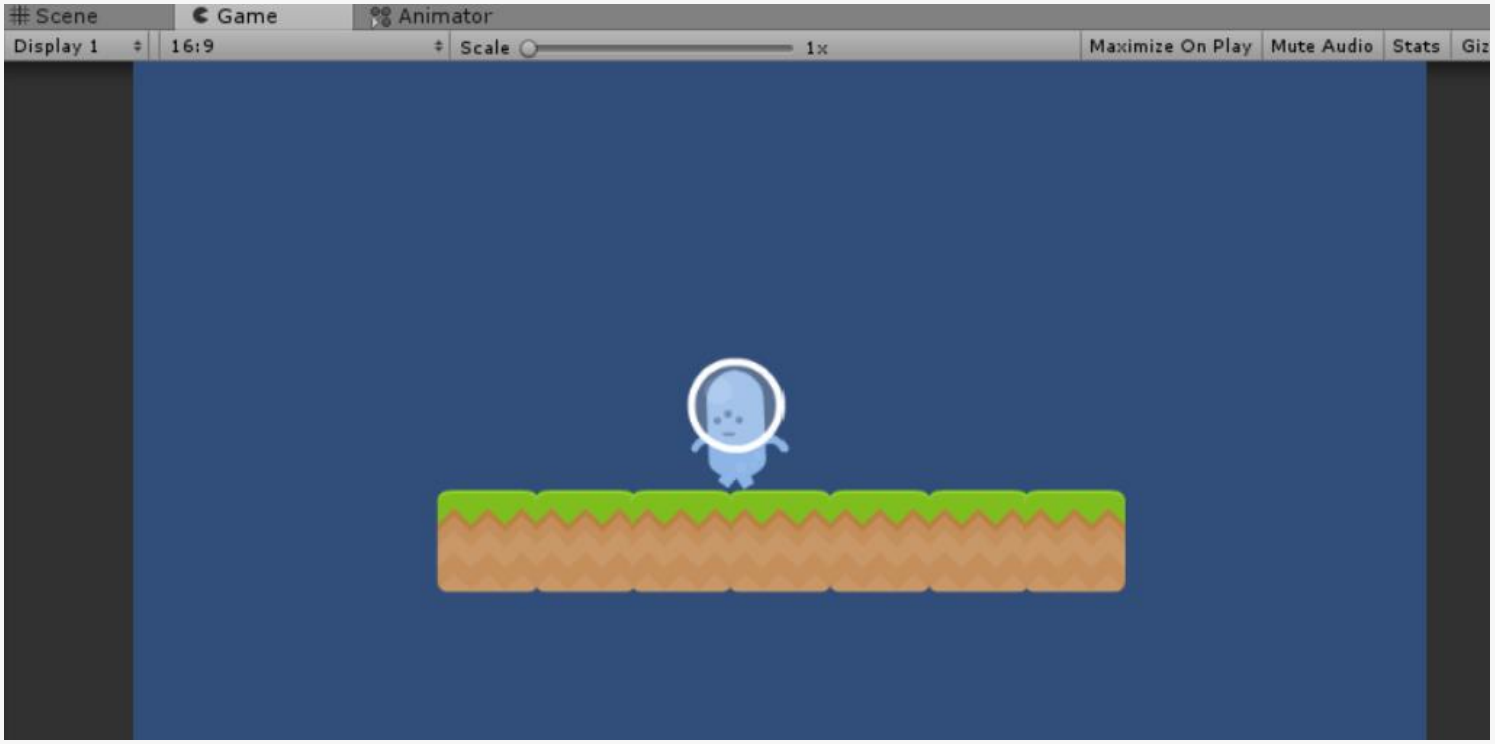
In this way, we flip the character according to its direction. Now, we need to take care of the Animator so it can change the animation accordingly. To do this, let's add this at the end of the Update() function

```
31          // Move along the object's x-axis within the floor bounds
32          if (transform.position.x + translation < rightWall &&
33              transform.position.x + translation > leftWall)
34              transform.Translate(translation, 0, 0);
35          // Switching between Idle and Walk states in the animator
36          if (translation != 0) {
37              anim.SetFloat("PlayerSpeed", speed);
38          }
39          else {
40              anim.SetFloat("PlayerSpeed", 0);
41          }
42
43      }
44  }
```

BINUS UNIVERSITY

People
Innovation
Excellence

Let's move!



BINUS UNIVERSITY

Assignment

Be more creative!!

Try to add more animation !!!!

People
Innovation
Excellence

BINUS UNIVERSITY

People
Innovation
Excellence

Using another key?

```
using UnityEngine; using System.Collections;

public class ExampleClass : MonoBehaviour {
void Update()
        {
                if (Input.GetKeyDown(KeyCode.Space))
                print("space key was pressed");
        }
}
```

BINUS UNIVERSITY

People
Innovation
Excellence

References

Freeman, J. (2015). Unity's New 2D Workflow

Vidyasagar. (2014. Unity and C#: Game Loop.CodeProject

Pereira, V. (2014). Learning Unity 2D Game Development by Example. Packt Publishing, Inc. San Francisco. ISBN: 9781783559046