In the “Mario Assets” folder, you are given *sprites*, or 2D images that will be used for your mario game.

A basic Mario game will include:

Animated Mario character

Animated turtle enemy

Functioning question mark block

Decoration (clouds, bushes)

To import these sprites, drag them into your Assets folder, or go Assets → Import New Asset…

After importing each sprite, click on it in the Project View and change its Pixels Per Unit value to 16 (from 100). This will create the “16 bit” effect for the game.

By dragging the sprites into the scene you are able to create a new GameObject with a SpriteRenderer component already set to display that sprite.

For the Mario and Turtle sprites however, you will have to split into multiple sprites.

To split a sprite, select it and change the Sprite Mode to Multiple.

Click the Sprite Editor, at the top left select Slice and set the grid to 16x by 16y.

This will split the sprite into 7 different sprites, which can now be turned into animations.

Select babymario\_0 and babymario\_1, and drag them into the scene. You will be prompted to save your new animation, create a folder named “babymarioAnimation” and save it there.

This will create a GameObject with an Animator controller that contains a brand new animation, babymario\_0 and babymario\_1.

Repeat this for babymario\_2 to babymario\_4 and babymario\_5 to babymario\_6 (creating three animations in total).

You only need one Animator Controller, so delete the two of the GameObjects and drag the missing two animations into the Animator View (accessed by double-clicking the Animator Controller).

The animator will need a **float** named “Speed” and a **bool** named “Jumping”

Using the given MarioController script, you will need to set up transitions from “Idle” to “Jump” and “Run”, from “Run” to “Jump”, and “Idle”, and from “Jump” to “Idle”.

It should transition to “Run” if “Speed” is greater than 0, and if “Jumping” is false

It should transition to “Jump” anytime “Jumping” is true

It should transition to “Idle” when “Speed” is 0 and “Jumping” is false.

Add the SpriteRenderer, AnimatorController, Rigidbody2D, and MarioController to your Player GameObject to complete it.

As for the Stone and Pipe GameObject, they should have a SpriteRenderer and a BoxCollider2D component. The BoxCollider can be resized, and it is what determines the collision boundaries of the object.

If you have completed all of this, you should have a basic Mario game with lots of room to add your own parts, which we will be covering in later lessons.

For now you can: make it so the player can’t walk off the left side of the screen, animate the turtle enemy (turtle\_0 and turtle\_1 are the walk, turtle\_2 and turtle\_3 are the death animation), or even add your own things (make sprites in paint, 16 by 16 pixels).