Contents

[What’s different? 1](#_Toc10538745)

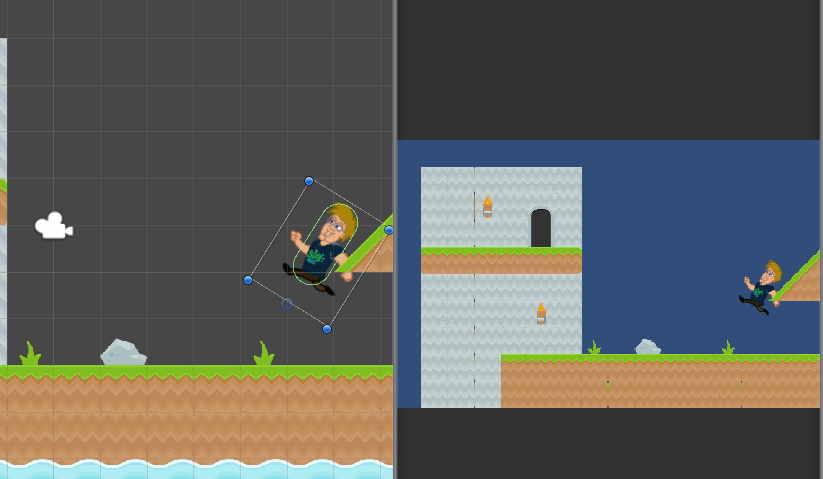
[Colliders 2](#_Toc10538746)

[Box Collider 2D 3](#_Toc10538747)

[Circle Collider 2D 3](#_Toc10538748)

[Edge Collider 2D 4](#_Toc10538749)

[Polygon Collider 2D 5](#_Toc10538750)

[Rigidbody 2D 6](#_Toc10538751)

[Colliders with the Tile Editor 7](#_Toc10538752)

[Object Physics 9](#_Toc10538753)

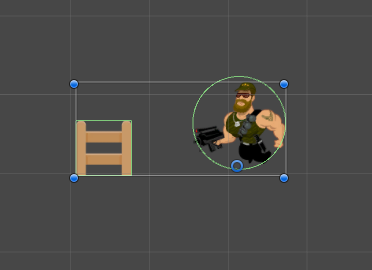
[2D Physics in Action! 10](#_Toc10538754)

# What’s different?

3D objects can still appear in a 2D Unity game. However, we are most likely going to be placing 2D objects into our scenes for a 2D game.

As such, we probably don’t want to be putting sphere and box colliders on our objects, since those are strictly 3D used for 3D collisions and we don’t have to worry about that 3rd direction.

To help with that, Unity provides plenty of 2D alternatives for all things Physics!



## Colliders

The most fundamental 2D physics components that Unity provides are Colliders.

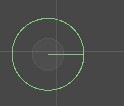
The common 3D colliders are: Box, Sphere, Capsule and Mesh.

The common 2D colliders in Unity are: Box, Circle, Edge and Polygon.

Let’s take a brief look at each of these.

### Box Collider 2D

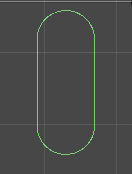
This differs from the 3D equivalent in that it doesn’t have that third dimension. If you have a rectangular shape that you want objects to bump into, this is the collider for you!

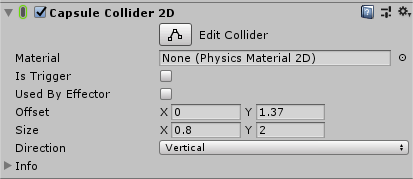


### Circle Collider 2D

Circle colliders are shaped like a circle…

These are a useful shape for not just circular things, but anything that’s roughly round. A rocky asteroid or a bullet could be usefully represented as a circle, depending on how accurate you’d like collisions to be.

Set the radius to create the size

  
Capsule Collider 2D

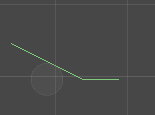
Capsules are useful for making the shape of a character. As it can be rounded but also stretched in a direction

### Edge Collider 2D

In 2D, we don’t always need to say that an area of the screen is occupied. Sometimes, we just want to have a wall or a floor. We can achieve this in Unity by adding an Edge Collider. This will create a line that objects can’t pass through. Simple. Effective.

To change the lines position, we open up the Points array

We can change how many points are in the Edge Collider by changing the size

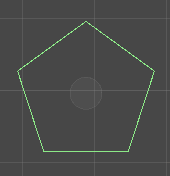
We can change there location by changing the Elements X and Y locations

Here is an example of the Edge collider 2D attached to an empty game object

### Polygon Collider 2D

Sometimes, we really need as accurate a collider as possible. When we have the need, the Polygon Collider will give us precise control over what we can and cannot collide with. However, these should not be used everywhere.

The Polygon uses the same Points system as the Edge, however this one closes up the gap between the last element and first element in the array.



## Rigidbody 2D

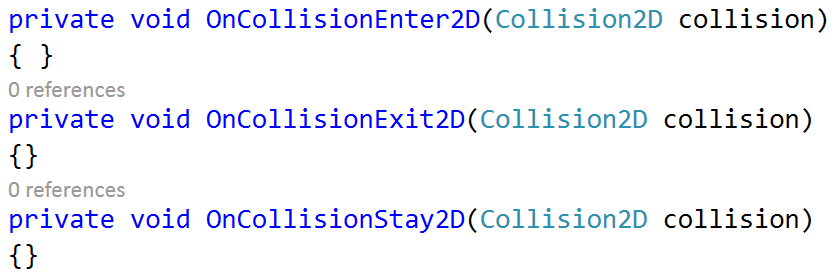
Just as Unity has a Rigidbody for 3D objects, there is also a Rigidbody for 2D objects. This behaves much the same as the 3D version, except you can also control how strong gravity will be for this object.

That’s pretty neat!

If you want use this for moving a character around, consider freezing the rotation. This can be found in the Constraints section at the bottom.

# Coding Physics

When coding with 2D physics, we have to take into account that we are using 2D assets and colliders. This means that our code is slightly different

* We call the Rigidbody2D Class instead of Rigidbody
* There is a special set of Collision functions for 2D collisions
  + As well as access to a 2D collider
* The same goes with 2D Trigger volumes as well

# Colliders with the Tile Editor

With our tilemaps, we can set them up to work with Unity’s physics systems.

We need a way to identify which tiles are collidable, knowing which spaces are drawn on.

Using Tilemap Collider 2D we can set up a tilemap to have all its tiles collidable.

1. Select the Tilemap you want colliders on

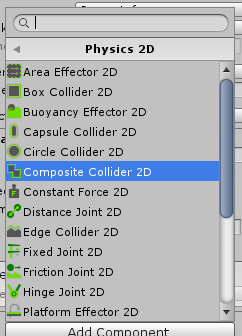
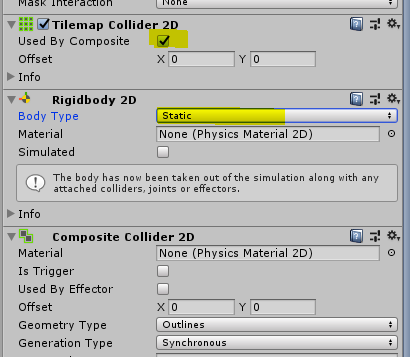
* In the Inspector

1. Click Add Component
2. Select Tilemap list
3. Select Tilemap Collider 2D

You will see that each tile in that tilemap will have a green shape around it.

This is good for when we need to Identify each tile individually, for pathfinding or raycasting for example.

If we don’t need to worry about each tile, It would be good to join connected colliders together

In the Inspector

1. Click Add Component
2. Select the Physics 2D
3. Select Composite Collider 2D

This will add both the Composite Collider 2D and Rigidbody 2D

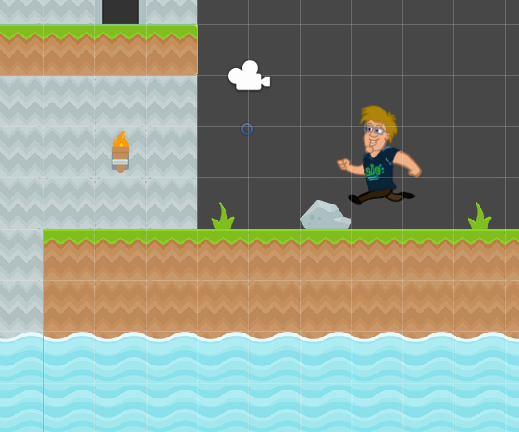
Now, we want to set it up so that the Rigidbody 2D is not moving, and that the Tilemap Collider uses the Composite collider

* In the Tilemap Collider 2D

1. Click Used by composite

* In Rigidbody 2D

1. Set Body Type to Static

The terrain should stay in place when the game plays, but also the tiles should have a line going all around instead of for each tile

# Object Physics

Now that we’ve learned what 2D physics components we have to work with, let’s put them to use.

* In your tile mapped level

1. Place one sprite from the hero sprite sheet into the scene
2. add a Capsule collider 2D to it.
   1. Resize the capsule so it fits the body
      1. Size x 0.8 y 2
      2. Offset Y 1.37
   2. Arms and legs can stick out a little
3. Add a Rigidbody2D to the character sprite.
4. Position the character sprite over a couple of the tile sprites.

# 2D Physics in Action!

When you press play the player will fall and collide with the scene.

Move the character in the scene during play mode to see changes to the physics.