Rube Goldberg Machine

Here we will talk more about the Rigidbody and Physics in Unity.

Rigidbody

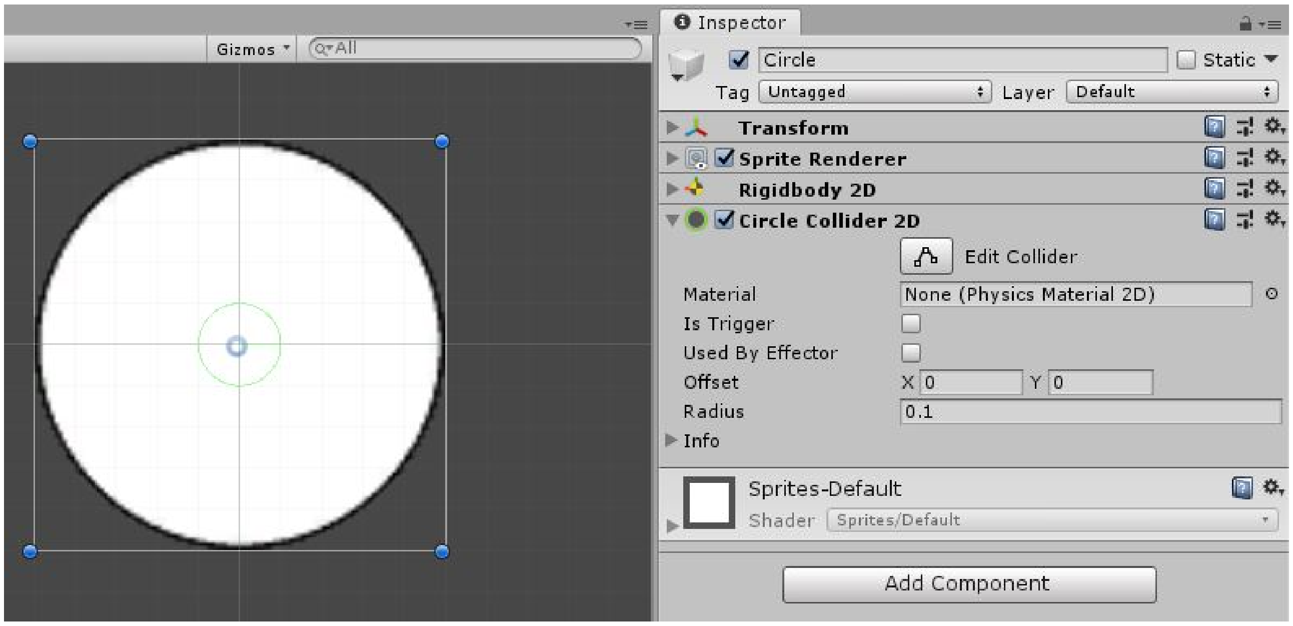
In the last lesson we added a Rigidbody to our circle and saw it fall. Let's talk a bit more about how this Rigidbody works.
The basic idea behind the Rigidbody is that it gives an object a physical presence in the world. Just like how you have mass and therefore are affected by gravity. If you didn't have mass you wouldn't be affected by gravity.
There are many other things the Rigidbody, such as apply drag to objects, but we are just using the basics of the component for this project.

Collisions

Collisions are one of the most important part of making your game work so let's get some going.

Colliders

In order for our objects to collide we need to give them a Collider. This allows an object to collide or hit other collider Objects.
Colliders are added to an Object the exact same way as Rigidbody. This is because a Collider is yet another Component.
Click on our first Circle Object and click add component and begin typing "Collider" until "Circle Collider 2D" shows up. Click on it.


Inspector
Circle
Static
Tag Untagged
Layer Default
Transform
Sprite Renderer
Rigidbody 2D
Circle Collider 2D
Edit Collider
Material None (Physics Material 2D)
Is Trigger
Used By Effector
Offset X 0 Y 0
Radius 0.1
Info
Sprites-Default
Shader Sprites/Default
Add Component

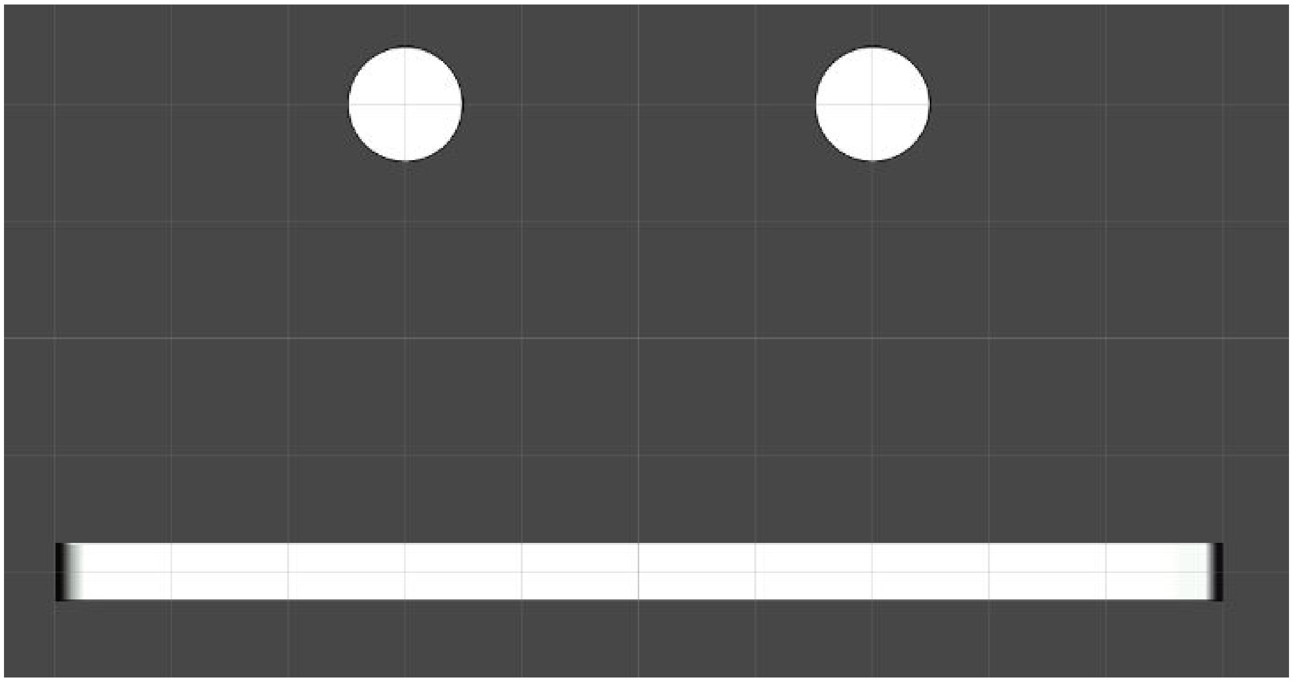
Note, keen eyes might've noticed my Inspector looks different. We've simply collapsed the components to reduce visual clutter. Click the gray arrow in the top left corner of the component.

You will now see there is a green circle within your circle. This is what the Collider considers its edge.
Since this green circle is too small we need to change the size of it. You can click on the "Edit Collider" button and drag the green squares to change it or in this case set the Radius to "0.5".
You can tell if the collider is the right size by simply seeing if the size of the green circle is close to the size of the white circle. In order for us to see the Collider in action we need something else to collide with.

- Add a new Sprite to your scene.
- Give it a name like "Ground".
- Set it's position to "0, -2, 0".
- Set it's Scale to "10, 0.5, 1".
- Add the "WhiteBox" file from the Assets folder to the Sprites folder in the Project. This is just like the circle from last lesson.
- Set the Sprite to the "WhiteBox" sprite.

- Add a "Rigidbody 2D" to it.
- Add a "Box Collider 2D". This should be the proper size, but you can change it if not.

It should look like this :

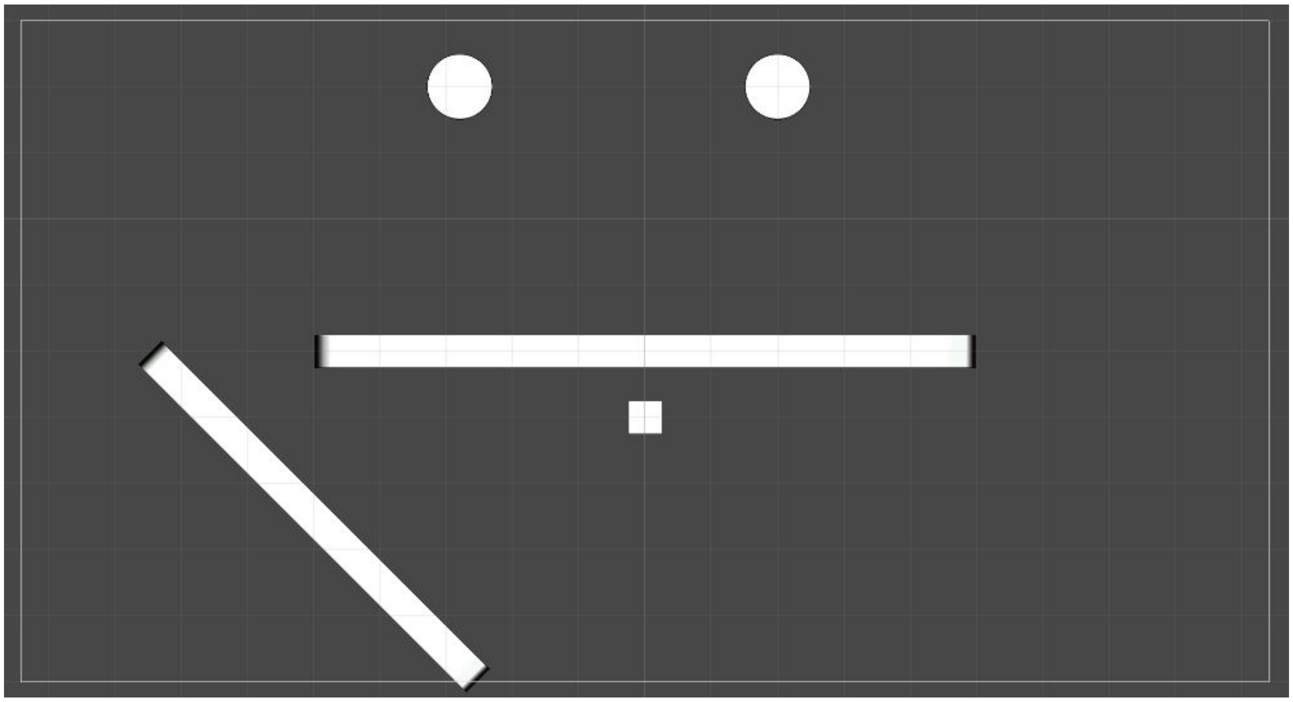


Now run your game!
Oh hold on, that was kinda lame. The box fell too. What good is ground if it falls too?
Let's fix this quick. We just need to change the Rigidbody "Body Type" of the box from "Dynamic" to "Static".
The "Body Type" tells Unity how to compute the physics of the Object. In our case Dynamic means the Object is affected by physics and Static means the Object is present in the "Physics Space" but not changed by interactions.

Let's run the game again.
Ok still kinda lame, but we're getting better!
Let's try one more thing.

- Right click on your box and duplicate it. Alternately use `Ctrl` + `D` or `Ctrl` + `C` and `Ctrl` + `V`.
- Rename it to something like "Anchor".
- Set the Y Position to "-3".
- Set it's Scale to "1, 1, 1".
- Finally set the Body Type of the first box (the big one on top) back to Dynamic.

Now run it!
We're cooking with gas now!
There's just a few more things we're going to do to make this a bit more interesting.
I encourage you to try setting up the world on your own now. Based on the picture below try and do this yourself. Don't worry about exact values, just see if you can make something similar.



If you can't get this setup just click below for the cheat sheet. But do try it on your own first!

▶ Scene Setup Help

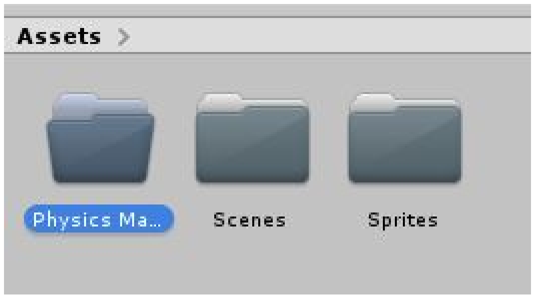
Physics

Unity has a built-in physics engine to make all of your game objects behave realistically.
Rigidbodies and Colliders are key parts of the physics in Unity but don't strictly need to be affected by the physics.
We're now going to talk about how you can change the way physics behave by using Physics Materials.

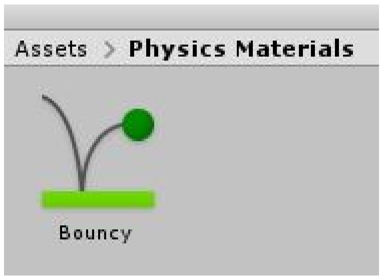
Physics Materials

In Unity, we can give our Rigidbody a Physics Material. These affect the way physics are calculated on our Objects. We can change the bounciness and friction of our elements. There are other things we can do but these are the most basic and useful to use.
Just a note, Physics Materials and Materials are different things, but we will be referring to Physics Material here.

We need to make a Material first. To up hold our current organization structure, start by making a new folder in our Project call it "Physics Materials". Just like we did before with the Sprites.

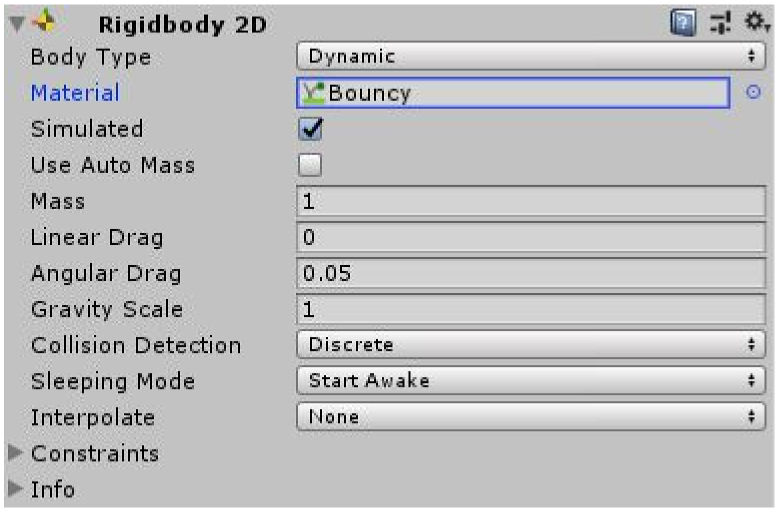


Navigate into that folder, right click, go to create, and then find and click on "Physics Material 2D" under the create section. Click on that to create a new material.
Now name that Material "Bouncy".



Go and click on the new Material. If you look over at the Inspector it's likely you can guess what we are going to change.
That's right! Set the "Bounciness" property to 1. It can be whatever you'd like but it's best not to go too high.
When we change the "Bounciness" of a material we increase or decrease the amount of energy retained after each collision.

Go and put that Material, "Bouncy", onto our first circle.
Click onto the Circle and then into its RigidBody. Click the setting named "Material", and choose our new Material.

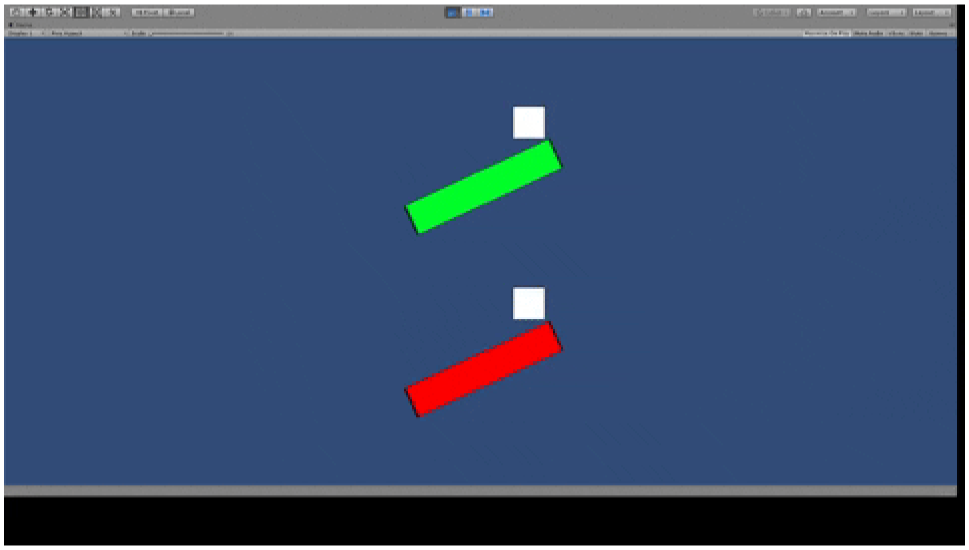

Rigidbody 2D

Body Type	Dynamic
Material	Bouncy
Simulated	✓
Use Auto Mass	
Mass	1
Linear Drag	0
Angular Drag	0.05
Gravity Scale	1
Collision Detection	Discrete
Sleeping Mode	Start Awake
Interpolate	None
Constraints	
Info	

If you play the game now you will notice the first circle bounces much higher.
Go ahead and play with the "Bounciness" value until you find something you like.

The other property on the Physics Material 2D is Friction.
The idea behind Friction is the same as the real world but the execution in Unity is a bit different.
Unity's friction is calculated based on the number of points of contact between two objects. Meaning that a circle isn't really affected by friction as technically there's only ever one point of contact.

Below is the effect in action. We've colored the high friction slope to be red for you easily identify.



If you'd like to get a feel for this yourself go ahead and set this up.
You can do it right beside what's already there in your project. You may need to move the camera though.
If you are having trouble with this click below for some more details. It's highly encouraged that you try and set this up on your own first.

▶ Setup to test friction

In the next lesson we are going to talk more about the Sprites we've been using and customizing them. Now we will get into more detail on how to manipulate them better as well as a bit more background on the idea. After that we are going to get to this units' project!