

# Easy Bike-Bicycle System

## Documentation and Information



***This bike/bicycle system is versatile and usable as a modular vehicle for a player to hop on and use.***

It works by using a physics-based Arcade approach that tends to emulate how a real-life bicycle would behave under normal riding conditions.

It offers versatile and easy access for 2-wheeled vehicle setups, which range from Motorbikes to Bicycles, allowing you to customize every ride to make it feel uniquely different. You can add your models and use this system on them to get your own custom assets running.

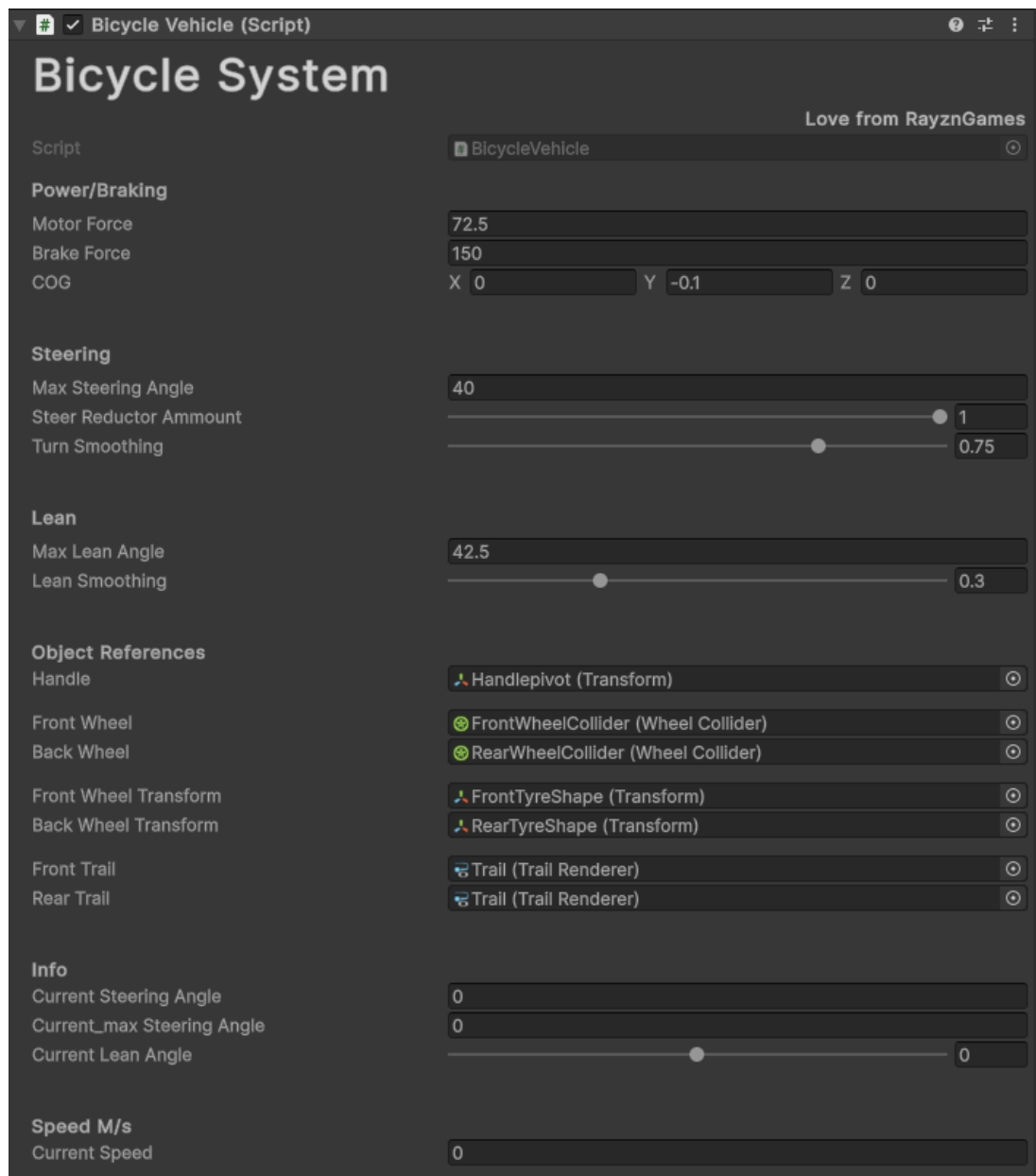
Documentation regarding how it works and how to use it can be found below.

The bike asset provided in the pack is usable for private and commercial projects. It is based on the CC Sharealike 4.0 International license

Bike asset made by me [Rayzn - Sketchfab](#)

**Thanks for downloading this Bicycle package!**

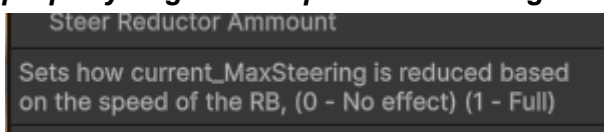
# How the package Works



The way the system works is an emulated physical approach, or SimCade system

Let's check all the properties one by one to know how to modify and create many different bicycle or motorbike behaviours.

***(Hover on top of any property to get ToolTip information regarding its setup)***



## Power and Braking

Power/Braking			
Motor Force	<input type="text" value="72.5"/>		
Brake Force	<input type="text" value="150"/>		
COG	X <input type="text" value="0"/>	Y <input type="text" value="-0.1"/>	Z <input type="text" value="0"/>

**MotorForce:** Applies a force directly to the rear wheel with no regard for gears or RPM. Cuts off power when Braking

**Brake force:** Defines the amount of braking applied to the wheels with no regard to Motorforce

**COG:** Defines a custom offset for the center of gravity point for the bicycle to remain more stable.(uses RB CenterOfMass + COG)

## Steering

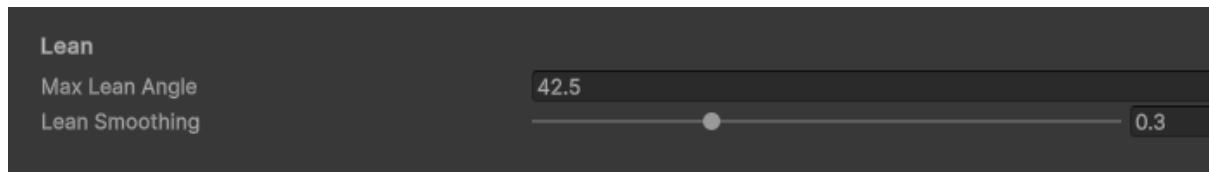
Steering	
Max Steering Angle	<input type="text" value="40"/>
Steer Reductor Ammount	<input type="text" value="1"/>
Turn Smoothing	<input type="text" value="0.75"/>

**MaxSteeringAngle:** Defines the maximum possible angle that the bike handlebars can reach

**SteerReductorAmmount:** defines the amount of reduction for the current\_maxsteeringangle when going at speed, reducing the steering angle the faster you go.

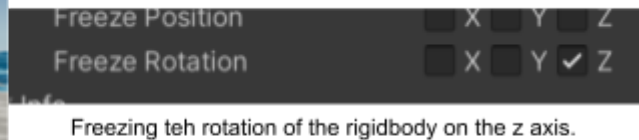
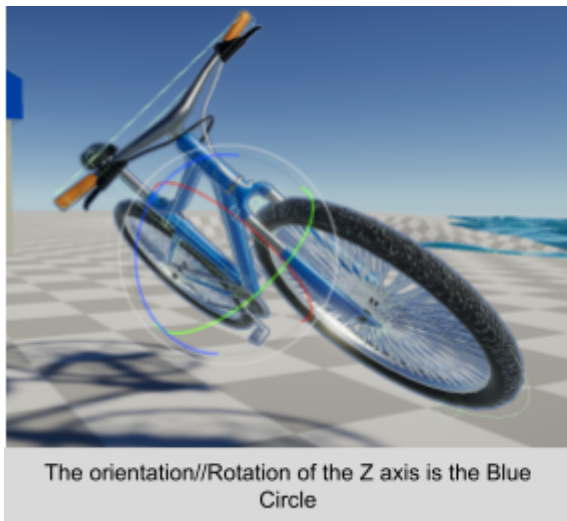
**TurnSmoothing:** Defines how fast the handlebars can reach the desired current\_max steering angle, making steering stiffer or softer

## Lean / Leaning



**maxLeanAngle:** Defines the maximum Angle in degrees of (lean/ tilt) reachable by the bicycle.















**leanSmoothing:** Defines how fast the bicycle can reach its maxLeanAngle



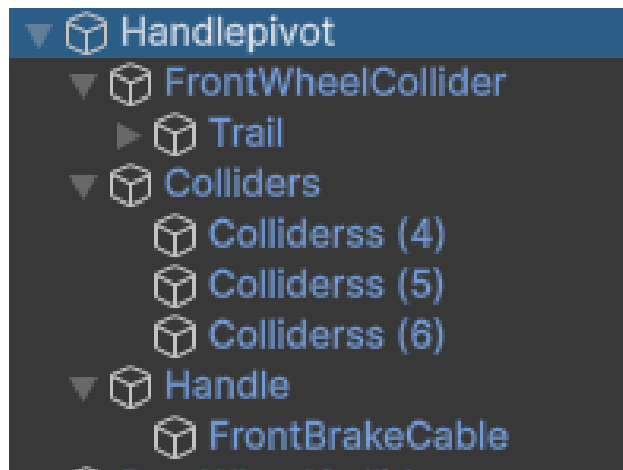
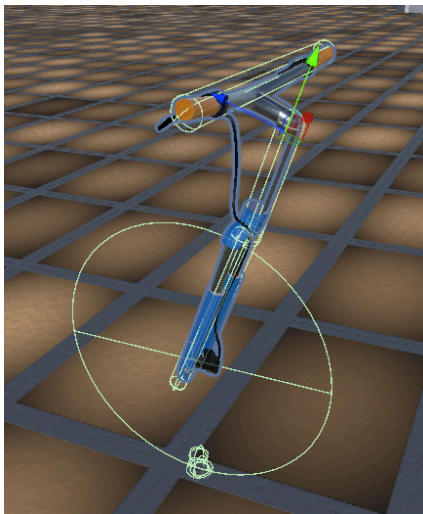
The main object of the bicycle has a Rigidbody that needs to have the Freeze rotation on the Z component of the RB Rotation, thus allowing us to take control over the tilt of the bicycle with the **currentLeanAngle** property.

## Object references:

All the important aspects and references for the bicycle objects are located here

Object References	
Handle	 Handlepivot (Transform) 
Front Wheel	 FrontWheelCollider (Wheel Collider) 
Back Wheel	 RearWheelCollider (Wheel Collider) 
Front Wheel Transform	 FrontTyreShape (Transform) 
Back Wheel Transform	 RearTyreShape (Transform) 
Front Trail	 Trail (Trail Renderer) 
Rear Trail	 Trail (Trail Renderer) 

**Handle:** Holds a Transform reference to the parent holder (**HandlePivot**) of the entire front fork that holds: handles and fork 3D models, and the front wheel collider.



**Front Wheel** Reference to front wheel collider

**Back Wheel** Reference to back wheel collider

**Front Wheel Transform:** Reference to front wheel visual

**Back Wheel Transform** Reference to rear wheel visual

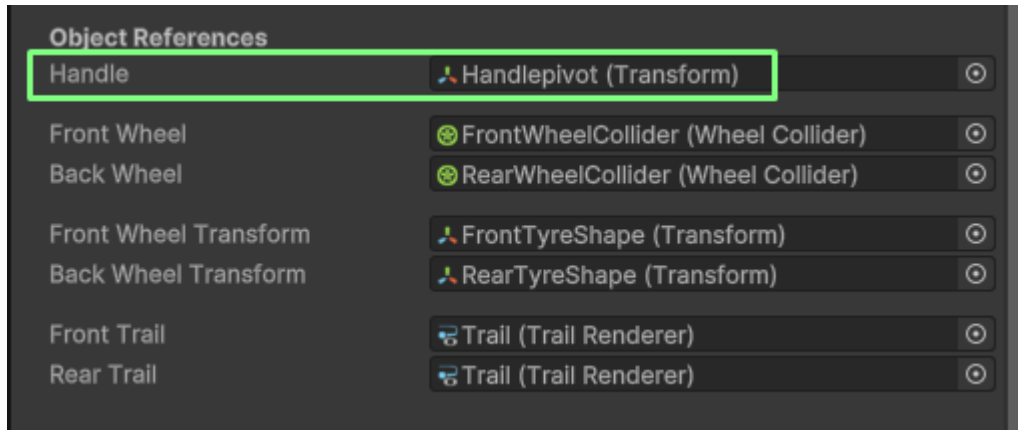
**Front Trail:** The trail renderer component of the front wheel

**Back Trail:** The trail renderer component of the back wheel

## To add your bike/bicycles Models:

- FrontWheelCollider should be parented under Handlepivot.  
**Handlepivot** should be the assigned transform in BicycleScript < **Handle**

(See [Handlepivot position and rotation in BicyclePrefab](#))

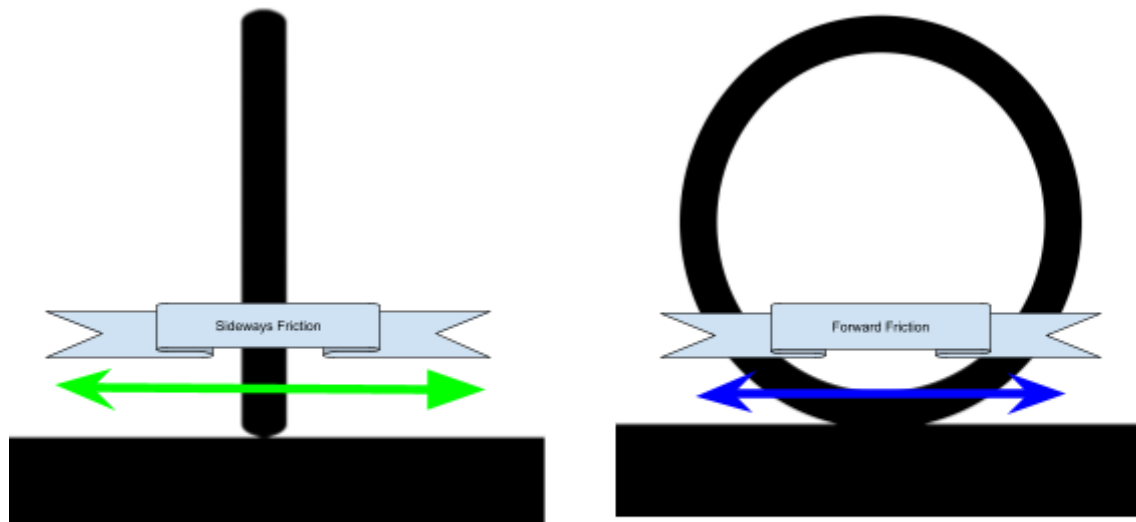


- **Wheel colliders:** should be placed separately from the tire transforms, but in the same position.
- **Suspension:** *(Currently configured suspension is physically accurate)*  
Spring force and dampening should be adjusted for proper suspension if you add more weight or remove it. Depending on the vehicle
- **Modify:** values like **motorforce**, **brakeforce**, **maxSteeringAngle**, **maxleaningAngle**, **turnSmoothing**, or **leanSmoothing**, modify the **weight of the rigid body**, and adjust the **size** and **wheelbase**, to achieve different bike configurations and behaviors. Allowing you to make multiple vehicles behave differently.

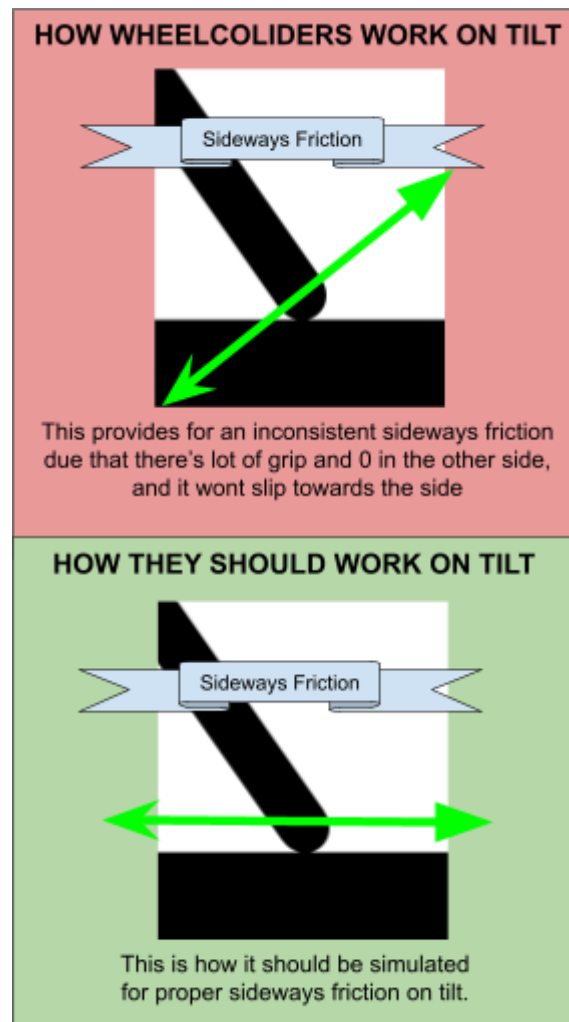
A video regarding how to prepare and import a model or bicycle to be imported and used will be available soon. (Possibly a model importer Setup tool)

## Possible issues to keep in mind!

WheelColliders are designed for vehicles with 4 wheels in mind, as the slip values are not correlated with the rotational values in the Z-axis of the wheels, and only account for friction values related to the Y-axis and X-axis.



If the wheel is properly straight up, the friction values are calculated properly, and the tensor axes are used in accordance as the pictures above explain. Anything outside of this behavior, is excluded from the simulation and, actually there's no problem with this if you're making vehicles, with 4 wheels. whereas no wheel is going to tilt in any direction.



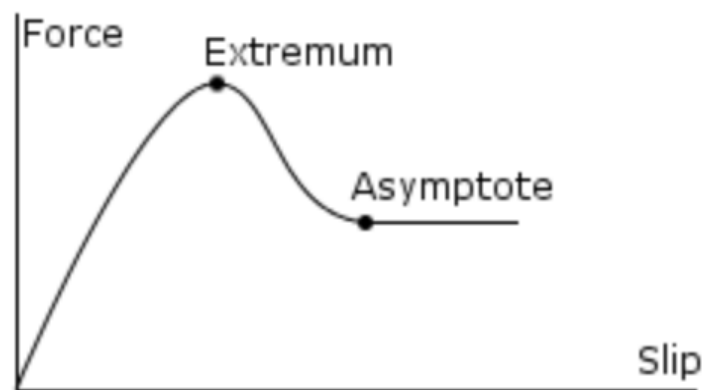
The problem with the wheel collider's friction properties is that they do not apply any friction value to the tilting. Maybe the problem resides in the fact that the sideways friction should be world-dependent instead of local axis-dependent.

The wheel collider as well, has got only one contact point, raycasting downwards, which creates several inconsistencies when on bumpy terrain.

This, plus the friction inconsistencies, leads to sometimes unexpected bike behaviours, for example, when turning, fast under slow speed,s they twist fast in the turning direction due to the back wheel lifting the ground for some reason, claiming 0 grip and sending the bike on an aerial slide. That ends up with the player facing the wrong direction. I saw that similar thing in GTA 4 bikes

We could also limit the tilt to a less visible tilt and allow for the tire frictions to be smoother and less janky or even remove the tilt entirely to ensure full proper friction values. but slightly lowering them until you find a sweet spot where it tilts and doesn't lose enough verticality to maintain proper friction.

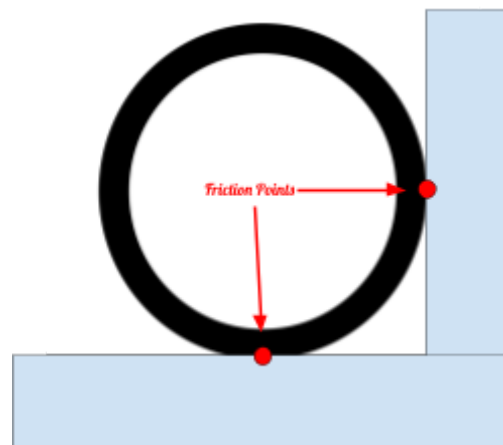
To properly set up the values for the grip levels, we had to understand the grip curve of a tyre to replicate the behaviour. Thankfully, Unity Documentations already have an approximate usual tyre grip graph, for you to replicate and play around with.



Typical shape of a wheel friction curve

### Possible Upgrades

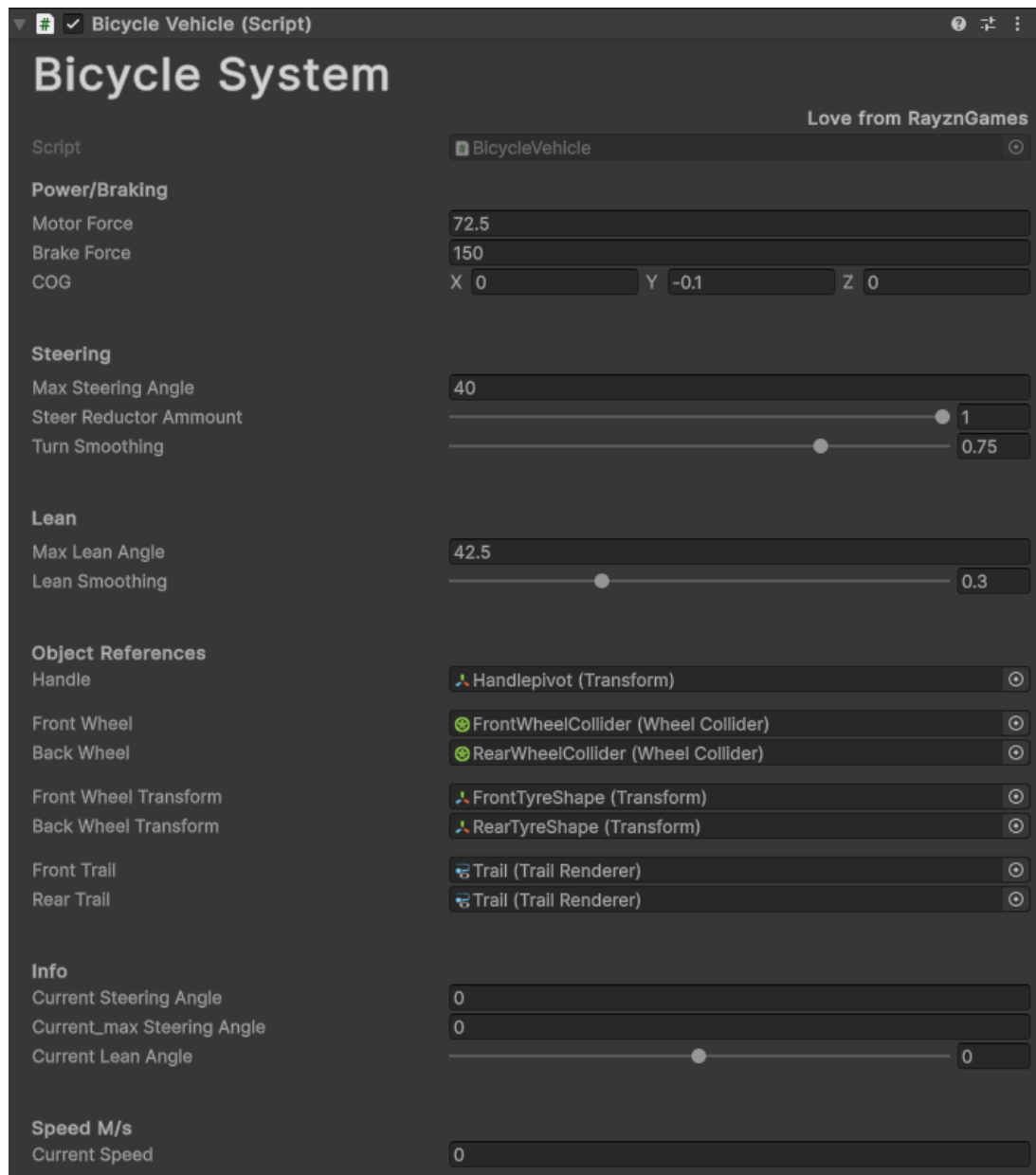
Custom colliders using the sphere cast function. The reason why sphere casting may be the way is that we may be able to stipulate multiple friction points of contact, even on surfaces where the wheel may experience friction from the down point, but the front point as well



But by now, the bike drives. and turns properly, tilting accordingly.

I'm going to leave it here and release the asset for free. I'm probably going to keep on working on them. But at least everyone can play with it.





Thanks for reading.- Sincerely