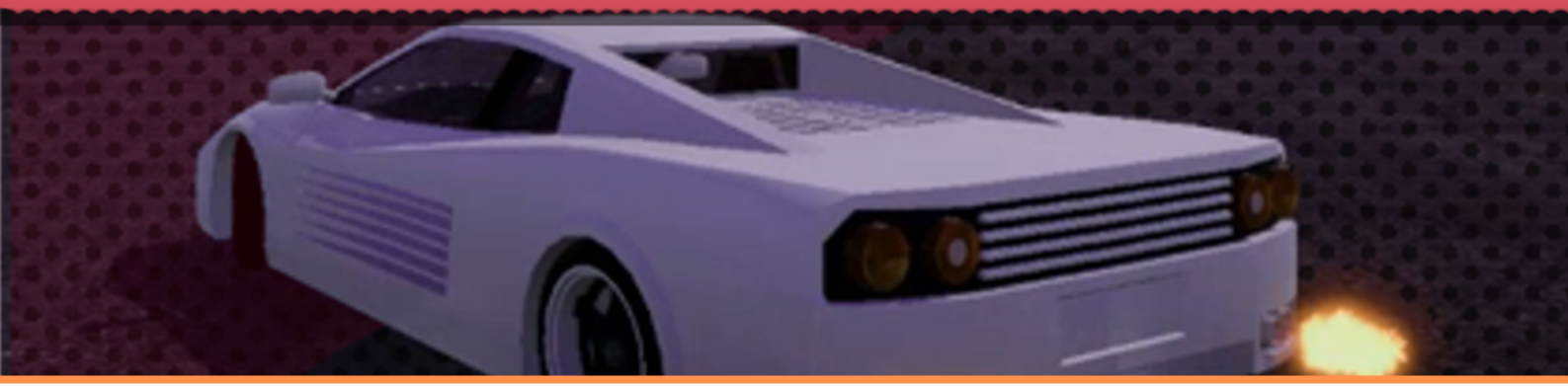
Custom Vehicle Controller



Overview:

CustomVehicleController is a Unity package for quick arcade vehicle physics integration into your project.

- Create a vehicle from different performance and handling parts.
- Create and customize parts from the custom editor window.
- Save parts presets.
- Initialize the controller in a matter of seconds.
- Supports runtime changes to any part, allowing you to tweak parameters during runtime and save changes after play mode.
- Easy part swap, allowing you to change performance parts at runtime.
- Scripts for handling engine and other vehicle sounds and different visual effects.

Package Contents:

1. Scripts for handling vehicle physics, sounds, and visual effects.
2. VisualEffectAssets and Particles Systems for the following visual effects:
    a. Tire Smoke.
    b. AntiLag.
    c. Nitrous.
    d. Wind Effect emitting from the vehicle body.
3. TrailRenderer prefabs for the following visual effects:
    a. Tire Skid Marks.
    b. Wing Aero Effect.
4. Audio Clips for 3 different engines, tire slip, wind noise, nitrous, anti-lag, and forced induction sounds.
5. Exemplary vehicle parts and presets, saved as scriptable objects.
6. Demo Scene.
7. 3D vehicle model.

Requirements:

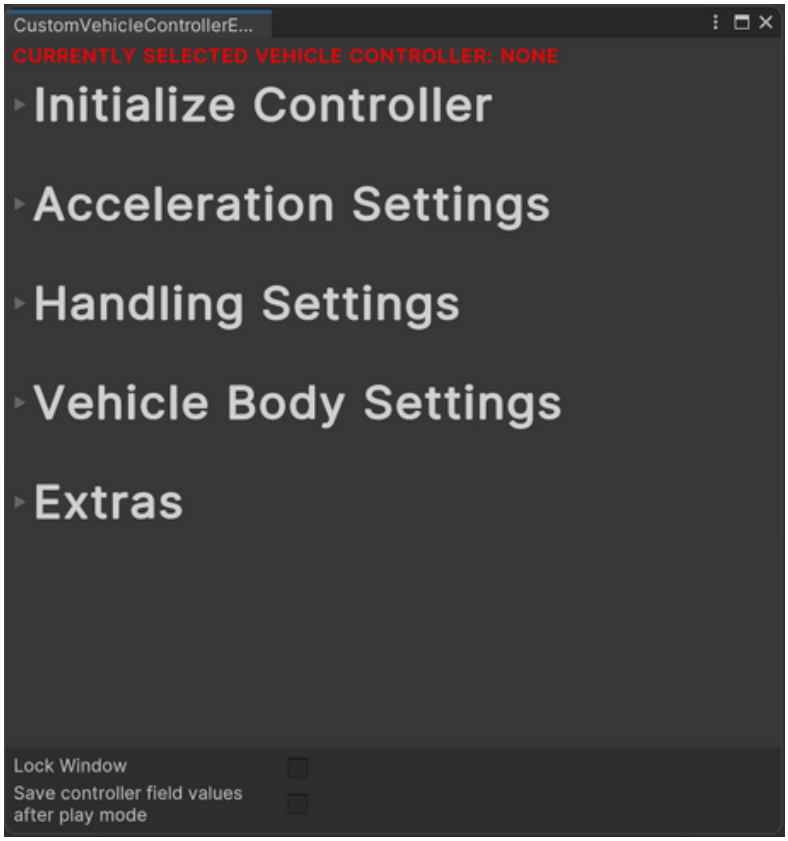
Unity version 2021.3 LTS or newer.
The demo Scene was created using the Universal Render Pipeline, meaning the materials and post-processing won't work in Built-In and High Definition Render Pipeline.
Visual Effect Assets require the VisualEffectGraph package installed.

Quick Start Guide:

1. Setting up the controller.

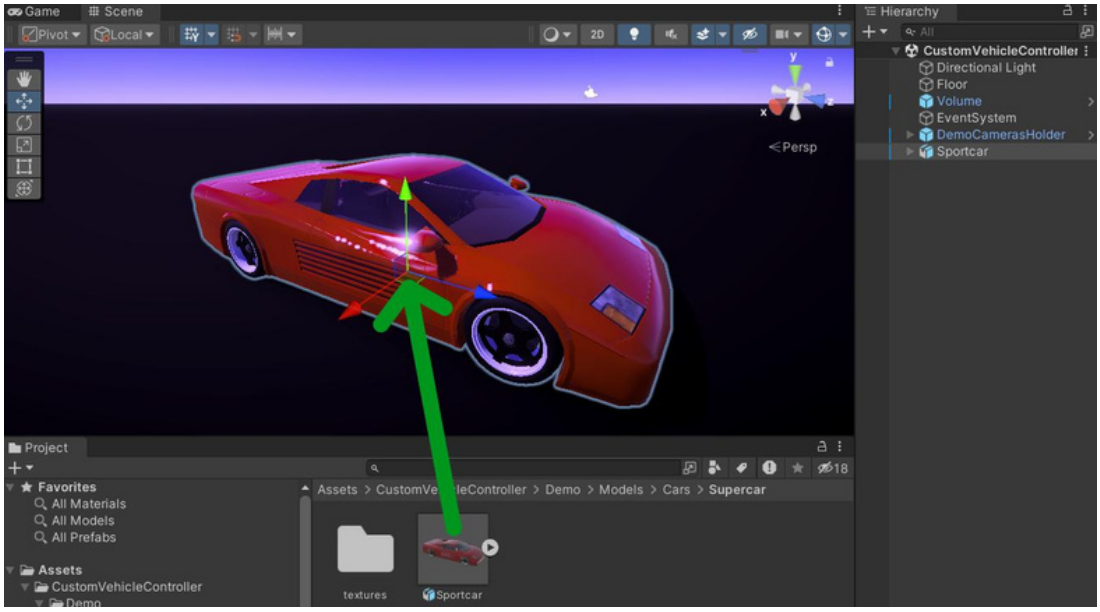
1. Open the editor window by going to Tools -> Custom Vehicle Controller -> Vehicle Editor.

An editor window should open:

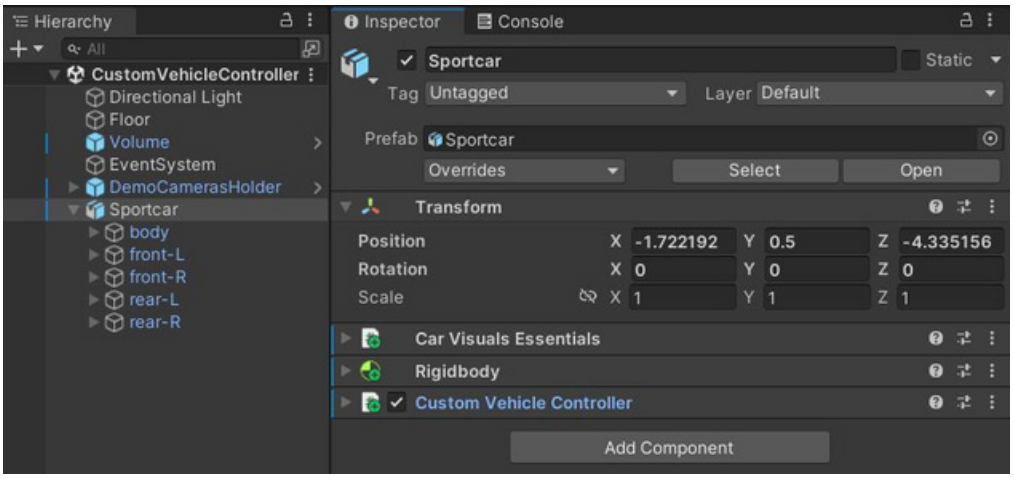


ⓘ For ease of use, you can dock the editor window.

1.1 Drag a vehicle model into the scene.

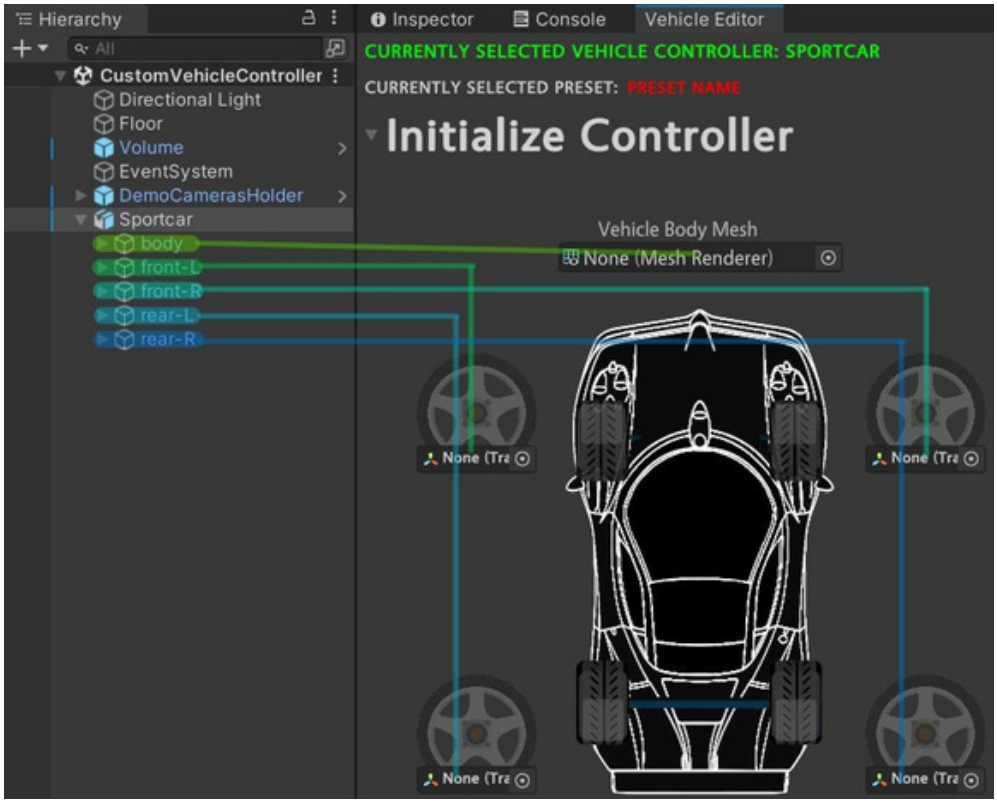


1.2. Select a root game object of your vehicle with the CustomVehicleController script attached.

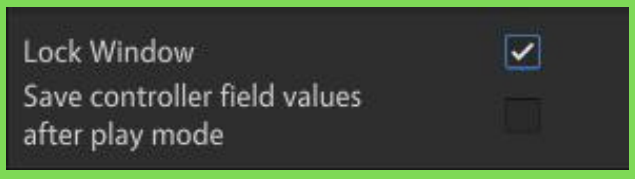


ⓘ If the selected game object doesn't have a CustomVehicleController component already, you can add it from the editor window.

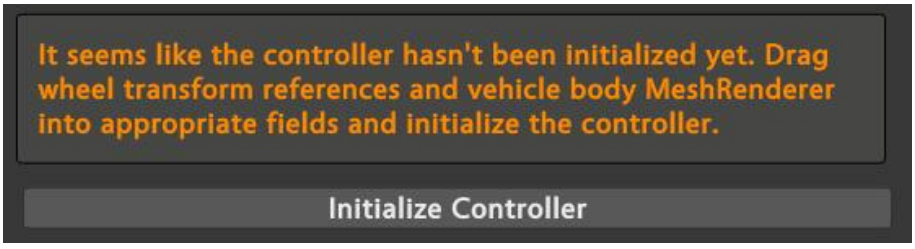
1.3 Drag and drop references to the wheels and vehicle body MeshRenderers.



ⓘ You can lock the window by checking the toggle at the bottom of the editor:

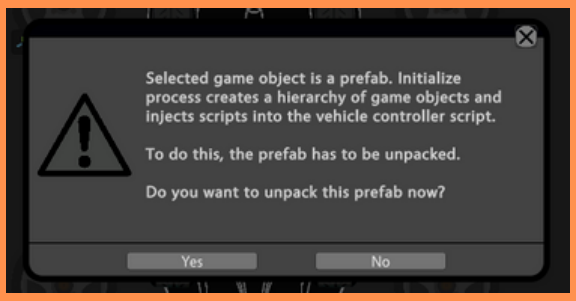


1.4 Initialize the controller.



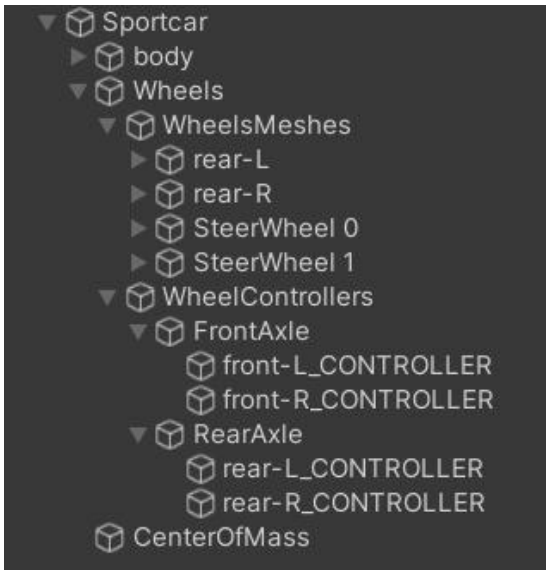
After clicking the "Initialize Controller" button, the hierarchy of game objects will be created, necessary scripts for the wheel physics added, script references populated, and, in case the wheel transform references have a MeshRenderer component, suspension position and wheel radius will be calculated.

ⓘ If the selected game object is a prefab, you'll need to unpack it. If you try to initialize the controller on the prefab, a warning will be shown:



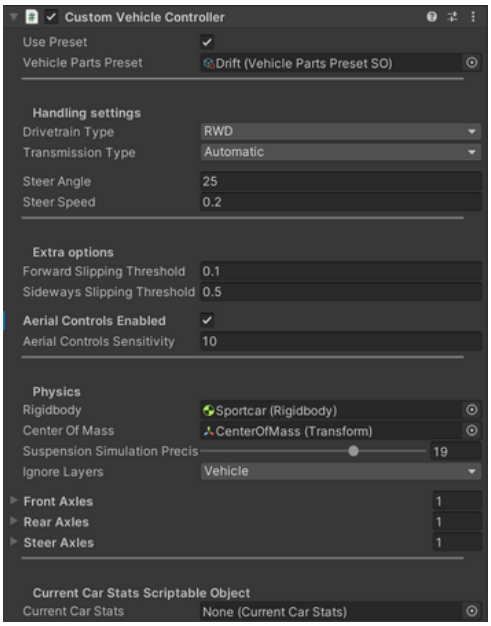
After clicking "Yes", the prefab will be unpacked completely and the controller will be initialized. You won't be able to initialize the controller as long as the selected game object is a prefab.

Now, the hierarchy should look like this:

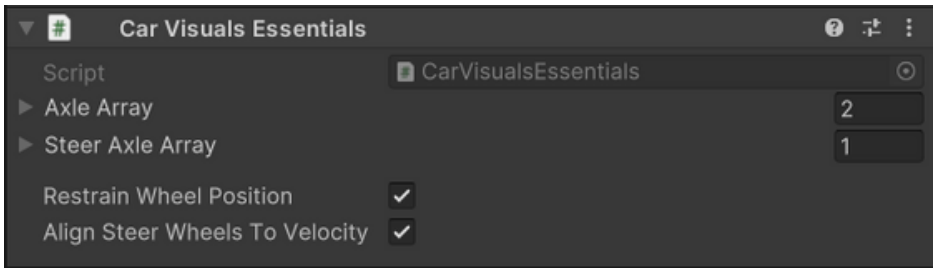


The selected game object should have:

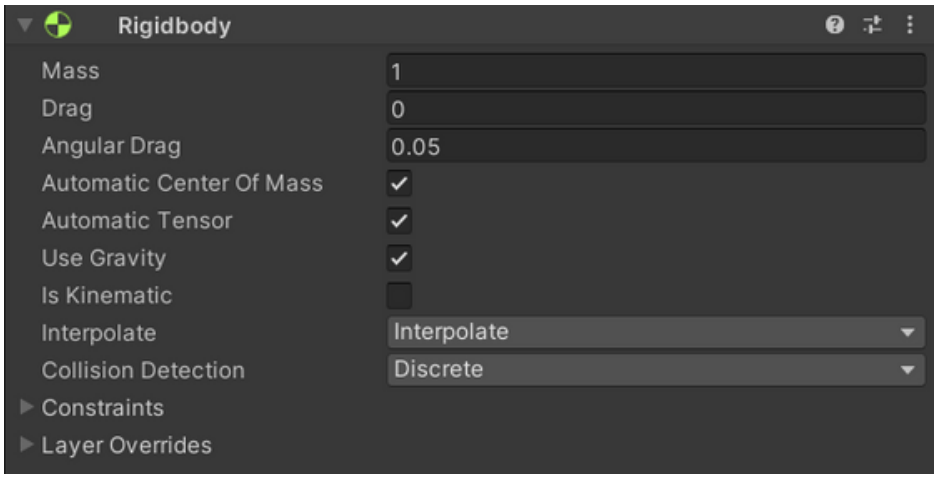
- CustomVehicleController component - manages the vehicle behavior.



- CarVisualsEssentials component - manages wheels` rotation and position.



- Rigidbody.



with references to newly created scripts and the center of mass assigned.

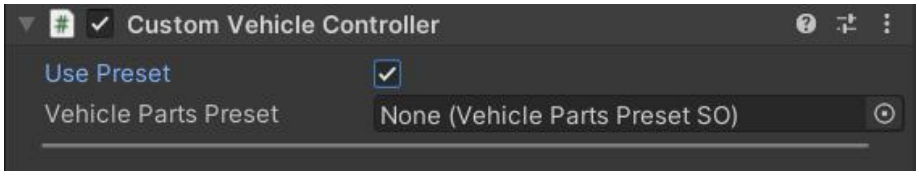
ⓘ For additional setup information visit online user guide.

2. Assign vehicle parts set.

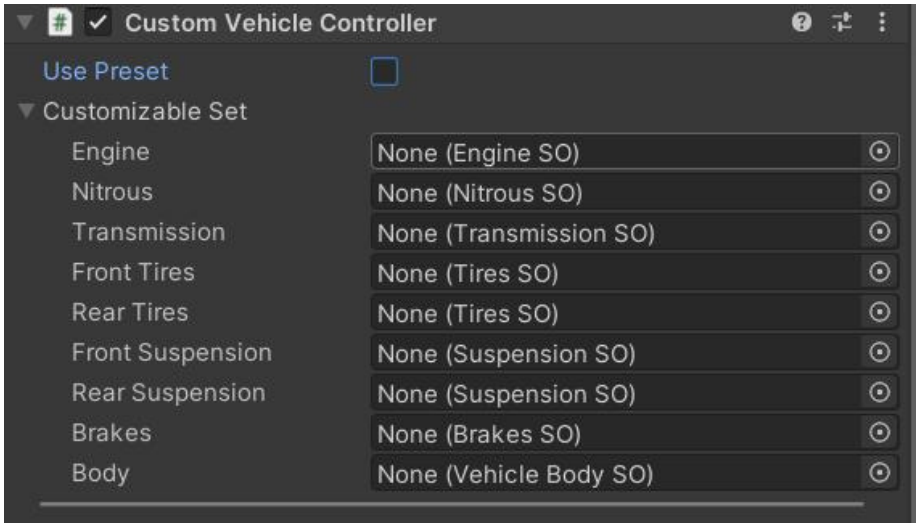
CustomVehicleController asset uses ScriptableObjects to represent the following vehicle parts:

- Engine.
- Nitrous (optional).
- Forced Induction (optional, part of the engine).
- Transmission.
- Tires.
- Suspension.
- Brakes.
- Body.

CustomVehicleController component expects either a VehiclePartsPresetSO, which is a scriptable object that holds references to the used vehicle parts:

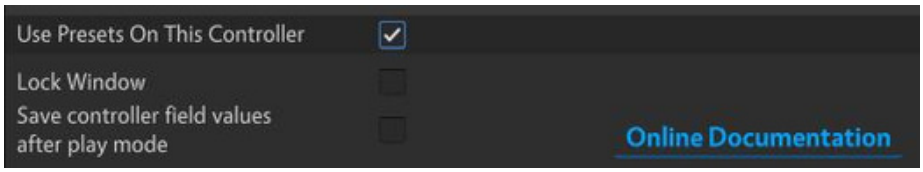


Or individually set up vehicle parts:

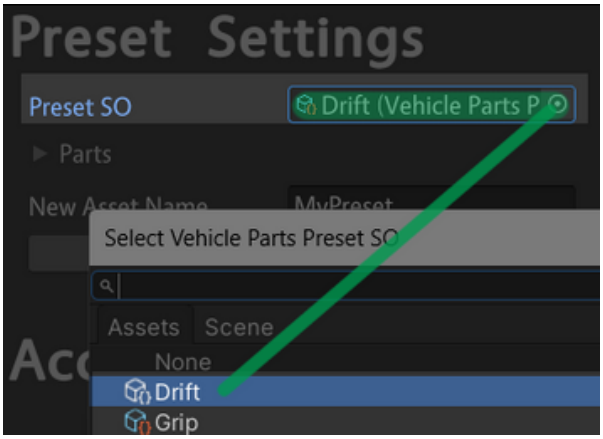


With the vehicle initialized and the vehicle game object selected, open the editor window.

Make sure the "Use Presets On This Controller" toggle is turned on:



In the Preset Settings foldout, select any VehiclePartsPresetSO instance.



ⓘ **For the vehicle parts preset creation guide visit online user guide.**

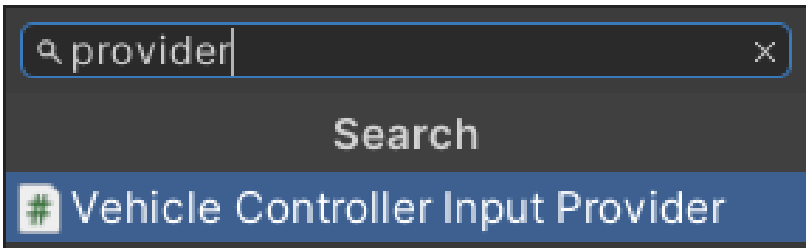
ⓘ **For the vehicle parts creation guide visit online user guide.**

ⓘ **For the vehicle parts guide visit online user guide.**

At this point, the vehicle should be set up completely. All that's left is input handling.

3. Player Input

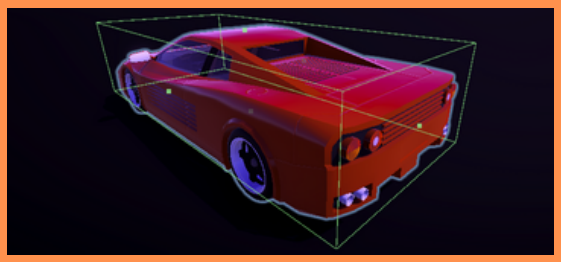
For handling player input, you are provided a script that uses old Unity's input system. Add it to the GameObject with the CustomVehicleController component.



ⓘ **CustomVehicleController expects a component that extends IVehicleControllerInputProvider. For more info visit online user guide.**

ⓘ **Do not forget to add colliders to the vehicle if you haven't already.**

If you use multiple colliders or a mesh colliders, additional set up may be required. Learn more from the online user guide.



You should have a car that drives.

For additional information, like how to create and modify parts, add sound and visual effects, expose current car stats, do advanced set up, etc., visit online user guide.