Table of Contents

---

1. Overview

`RCC_Inputs` is a simple data container used by Realistic Car Controller (RCC) to store **vehicle control** and **camera control** input values. It is typically populated each frame by the `RCC_InputManager` and then **read** by components like `RCC_CarControllerV4` (for driving physics) and `RCC_Camera` (for orbit and zoom).

By separating the input data into a single class, developers can more easily swap input methods (e.g., keyboard, gamepad, mobile UI) without modifying the vehicle or camera logic directly.

---

2. Class Declaration

```
[System.Serializable]
public class RCC_Inputs {
    // Fields...
}
```

- `[System.Serializable]` allows these fields to be visible and editable in the Unity Inspector (for debugging or custom tooling).
- This class is **not** a MonoBehaviour— it's a **plain C# object** designed for data storage.

---

3. Fields and Properties

Below is a breakdown of the fields within `RCC_Inputs`. All of them are **public**, making it easy for other scripts to read/write at runtime.

3.1 Primary Driving Inputs

1. **`float throttleInput`**
    - Range: **0 to 1**
    - **0** = no throttle, **1** = full throttle
2. **`float brakeInput`**
    - Range: **0 to 1**
    - **0** = no brake, **1** = full brake
3. **`float steerInput`**
    - Range: **-1 to 1**
    - **-1** = full left, **1** = full right, **0** = straight
4. **`float clutchInput`**
    - Range: **0 to 1**
    - **0** = clutch fully engaged, **1** = clutch fully disengaged
5. **`float handbrakeInput`**
    - Range: **0 to 1**
    - **0** = handbrake off, **1** = handbrake fully on
6. **`float boostInput`**
    - Range: **0 to 1**
    - Represents turbo / NOS usage. **0** = no boost, **1** = full boost
7. **`int gearInput`**
    - An integer representing the current gear selection.
    - Possible usage examples:
        - **-1**: reverse gear
        - **0 to 5+**: forward gears
        - **-2**: neutral gear
    - Actual meaning can vary depending on your specific gear system.

3.2 Camera Inputs

1. **`float orbitX`**
    - A horizontal axis, typically used by the camera system to orbit around the vehicle.
    - E.g., mapped to mouse X or right analog stick horizontal.

2. `float orbitY`
    - A vertical axis for orbiting the camera.
    - E.g., mapped to mouse Y or right analog stick vertical.
3. `Vector2 scroll`
    - A 2D vector typically representing zoom input (e.g., mouse scroll wheel, pinch zoom).
    - The camera logic decides how to use X or Y from this vector.

---

4. Usage Notes

- **Populated by RCC_InputManager**: Usually, you won't manually set these values. Instead, `RCC_InputManager` (or a similar input system) updates them each frame based on the current input device.
- **Consumed by Vehicle/Camera**: `RCC_CarControllerV4` reads `throttleInput`, `brakeInput`, `steerInput`, etc., to apply forces and torques. The camera scripts read `orbitX`, `orbitY`, `scroll` to rotate or zoom.
- **Range Clamping**: The `[Range(...)]` attributes in the code help communicate expected ranges, but they do not strictly enforce them. If you assign values beyond the specified range, there's no built-in clamp logic (unless you add it yourself).
- **Gears**: The `gearInput` field is optional depending on whether your controller system sets gears via an integer. Some systems may set this via separate events instead.

---

5. Example Usage

```
void Update() {
    // Suppose these values come from a custom input method
    rccInputs.throttleInput = Mathf.Clamp01(Input.GetAxis("Throttle"));
    rccInputs.brakeInput = Mathf.Clamp01(Input.GetAxis("Brake"));
    rccInputs.steerInput = Mathf.Clamp(Input.GetAxis("Horizontal"), -1f, 1f);
    rccInputs.handbrakeInput = Mathf.Clamp01(Input.GetAxis("Handbrake"));
    rccInputs.boostInput = Mathf.Clamp01(Input.GetAxis("Boost"));

    // Update camera orbit
    rccInputs.orbitX = Input.GetAxis("Mouse X");
    rccInputs.orbitY = Input.GetAxis("Mouse Y");

    // Then pass it to the RCC car
    myCarController.ApplyInputs(rccInputs);
}
```

> **Note**: In real scenarios, you'd typically use `RCC_InputManager` or `RCC_MobileButtons` instead of raw `Input.GetAxis`.

---

6. Summary

`RCC_Inputs` is a straightforward structure capturing **all necessary vehicle and camera inputs** in Realistic Car Controller. By consolidating these values, **input logic** is cleanly separated from **vehicle and camera logic**, making it easy to plug in different input sources (keyboard, mobile buttons, custom controllers) without changing the underlying driving or camera scripts.