Realistic Car Controller Pro

Thank you for purchasing and using Realistic Car Controller Pro. This documentation will guide you to definition of the **RCCP_SceneManager**.

Install & Import BCG Shared Assets To The Project

First, you'll need to import the latest BCG Shared Assets to your project. You can import it from the **Tools → BCG → RCC Pro → Welcome Window → Addons.** There will be two new scenes added to your scenes folder. Enter Exit FPS and Enter Exit TPS scenes.

Import Integration Package

Import the integration package from the **Tools → BCG → RCC Pro → Welcome Window → Addons**. Wait for a few seconds to compile the scripts. There will be two new scenes added to your scenes folder. Photon lobby and photon blank scenes.

How The Integration Works

FPS and TPS controllers are using different cameras. Usually, camera of the FPS character is placed in the gameobject. But TPS characters have unattached and independent cameras, and they are not parented to the character. Therefore, **BCG_EnterExitPlayer.cs** script has two options for usement with FPS or TPS. If TPS is choosen, you must declare the TPS camera in the scene. You can edit some settings from (**Tools → BCG → Shared Assets → Edit Settings**).

How The System Works (Simple)

All enter exit RCC vehicles must have "**BCG_EnterExitVehicle**" component. And our FPS/TPS characters must have "**BCG_EnterExitPlayer**" component. **BCG_EnterExitManager** (Will be created automatically) will be observing all vehicles and characters in the scene. If vehicle has no driver, *canControl of the vehicle will be disabled, and not registered as player vehicle. If our character has entered any vehicle, **BCG_ EnterExitManager** will register the target vehicle as player vehicle, enables the canControl of the vehicle. At that moment, our character will be parented to the vehicle and disabled entirely. Now, **BCG_EnterExitVehicle** has a **driver now. If player gets out from the vehicle, **BCG_ EnterExitManager** will be activating character controller, and transport it to "**Get Out Pos**" of the vehicle. And lastly, deregister the vehicle and disable *canControl of the vehicle. Now, vehicle is not controllable, not a player vehicle.

* = canControl bool of the RCCP_CarController. If it's enabled, the vehicle is controllable.

** = driver variable of the BCG_EnterExitVehicle.

How To Make It Work

All you have to do is, add necessary scripts to the vehicles and to your player. Get out position of the vehicles are created automatically in script if you don't select it in **BCG_EnterExitVehicle**. You can replace get out positions.

For FPS characters, it will depend on your camera of the FPS player. For TPS player, it depends on your actual TPS Player, not camera.

- Character Controllers must have **BCG_EnterExitPlayer**
- Vehicles must have **BCG_EnterExitVehicle**
- **_BCGEnterExitManager** will be created automatically in your scene. **_BCGEnterExitManager** will be observing and managing all the process.

How The System Works (Detailed)

Please read simple version first if you haven't read. Let's get it straigth with more detailed version;

1- **BCG_EnterExitVehicle** – must be attached to all vehicles.
2- **BCG_EnterExitPlayer** – must be attached to the character.
3- **BCG_EnterExitManager** – will be created automatically.

All the system is based on events. **BCG_EnterExitPlayer** is firing an event when it's spawned, destroyed, gets in a vehicle, or gets out of the vehicle. **BCG_ EnterExitManager** will be listening to
this event as well. So, manager knows your all of your character events. All events are explained below;

1. public delegate void onBCGPlayerSpawned(BCG_EnterExitPlayer player);

2. public delegate void onBCGPlayerDestroyed(BCG_EnterExitPlayer player);

3. public delegate void onBCGPlayerEnteredAVehicle(BCG_EnterExitPlayer player, BCG_EnterExitVehicle vehicle);

4. public delegate void onBCGPlayerExitedFromAVehicle(BCG_EnterExitPlayer player, BCG_EnterExitVehicle vehicle);

You can listen any events on the **BCG_EnterExitPlayer** or **BCG_EnterExitVehicle**.

For example;
BCG_EnterExitManager will use using **GetIn()** or **GetOut()** methods by listening these events. **GetIn()** method requires "**targetVehicle**". **GetOut()** method doesn't require any parameters.

When our character standing next to the any vehicle, get in text will be enabled by **BCG_EnterExitPlayer**. If player pushes the interaction button, **OnBCGPlayerEnteredAVehicle** event will be fired with **targetVehicle** data. As you well now, this event has been listening by **BCG_ EnterExitManager**. So, **BCG_ EnterExitManager** will be firing **GetIn(targetVehicle)** method to do
the rest…