Burnout Racers

Thank you for purchasing and using Burnout Racers.

Burnout Racers 1
Installation 2
Warning 2
Overview 2
- Settings 3
- Scene Templates 3
- Player Vehicles 4
- Bot Vehicles 5
- Enable / Disable Mobile Controllers 6
- Welcome Window 6

How to Add New Scenes? 7
- Adding a Main Menu Scene 7
- Adding a Gameplay Scene 9

How the Scenes Work? 9
- Main Menu Workflow 9
- Gameplay Workflow 11

Events 13
Photon RPC Events 14
How to Create and Add New Player Vehicles 15
How to Create and Add New AI Vehicles 15
How to Change AI Names 17
How to Change Player Vehicle Stats and Price 18
Performance on Mobile Devices (Shadows, and Post Processing) 19
- Realtime Shadows 19
- Post Processing and Profiles 20

Contact & Support 21

Installation

Please import the package to the new fresh project using **Unity 2022.3.26f1** or later version. Importing the asset to an existing project would override your current project settings, that's why a new fresh project recommended. After importing the asset, editor will search for **Photon PUN2** in the project and ask you to import if it can't find it.

Warning

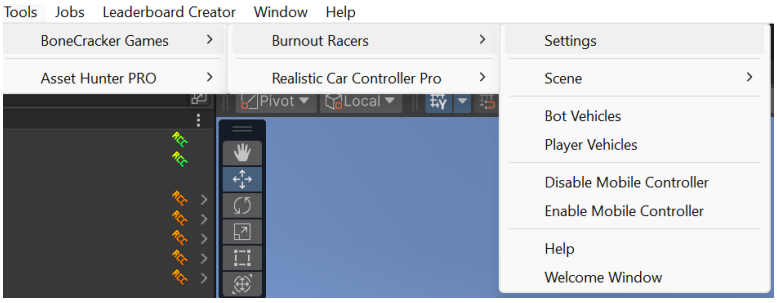
Project would still compile fine without Photon PUN2, however most of the game features won't be available. Please import the latest version of Photon PUN2 to the project, it will ask you to enter your own **AppID**. You can obtain your own AppID from Photon's website.

After importing the Photon PUN2 to the project, you shouldn't get any red errors on your console. Yellow warning messages about polygon count can be ignored, they won't be shown again. Open the demo scenes of the project and test it. Please don't open the gameplay scenes directly without logging in. Because if your latest game type selection was multiplayer / online mode, it will give you error. Default game type is multiplayer / online mode.

Overview

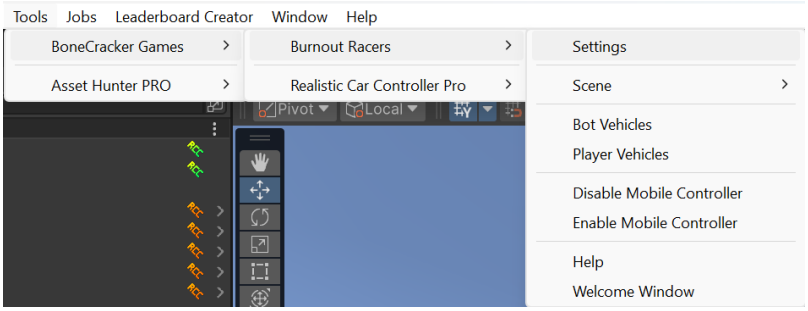
Project is using **Universal Render Pipeline (URP)**, Input System, Navigation, and other essential packages. These packages will be imported automatically. If your Unity version is the latest version, you may want to update these packages from the **Package Manager** (***Window → Package Manager***). Search for packages installed in the project and update them if necessary.

Project includes latest version of **Realistic Car Controller Pro (RCCP)**, you don't need to import it again. Also, you shouldn't update RCCP in the project, because Burnout Racers (BR) includes an edited version of RCCP. You can get access to the Burnout Racers settings from the ***Tools → BCG → Burnout Racers***. All topics will be explained below.



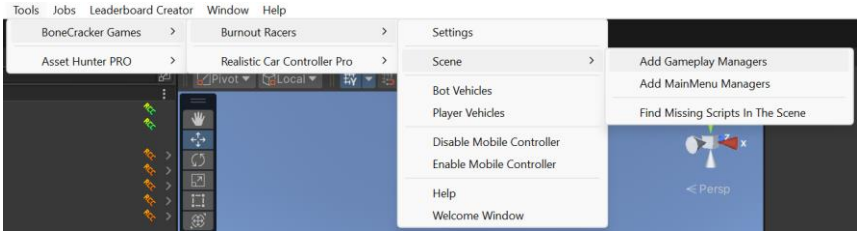
Settings

General settings can be accessed from the **Tools** menu. You can change initial startup money amount, soundtracks, optimizations, etc....



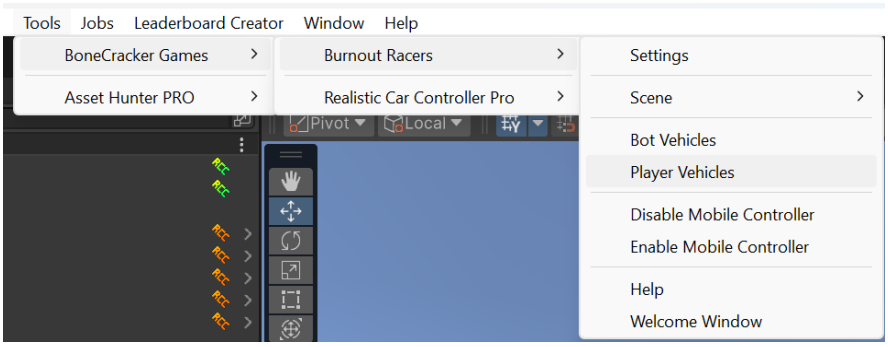
Scene Templates

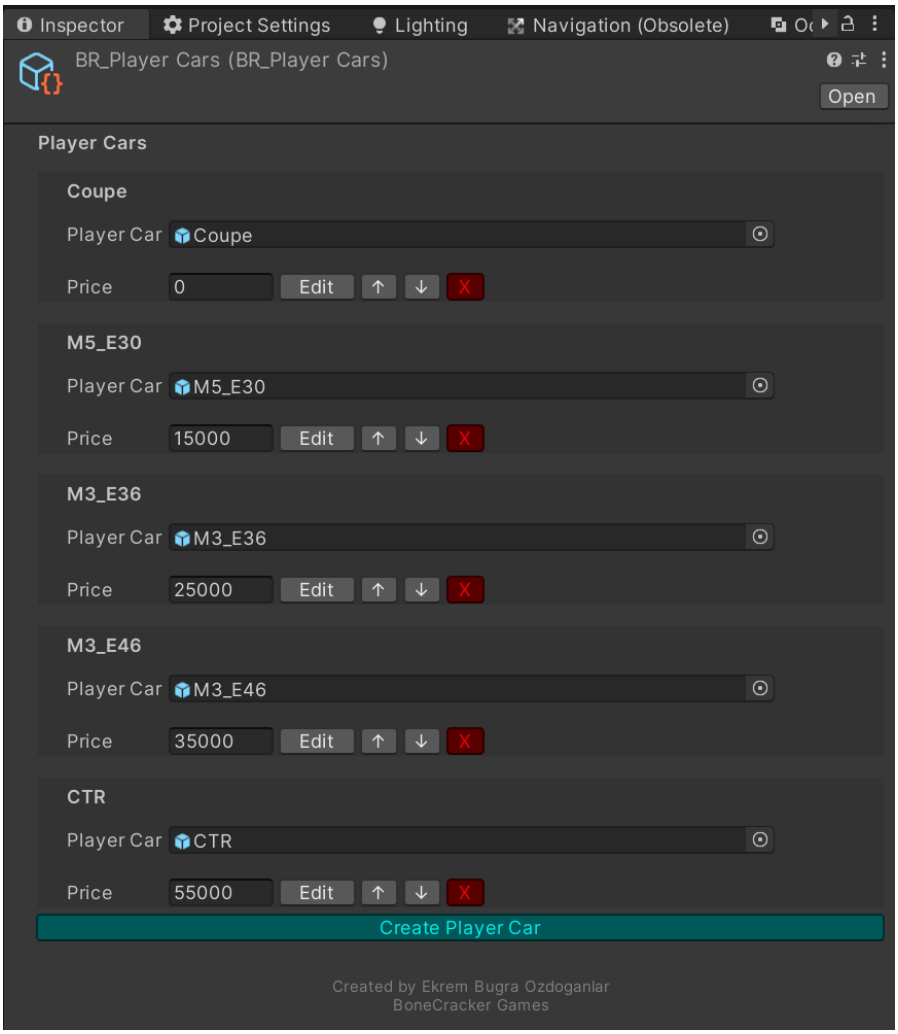
Scene templates can be accessed from the **Tools** menu. You can setup your own scene as a main menu scene or a gameplay scene.



Player Vehicles

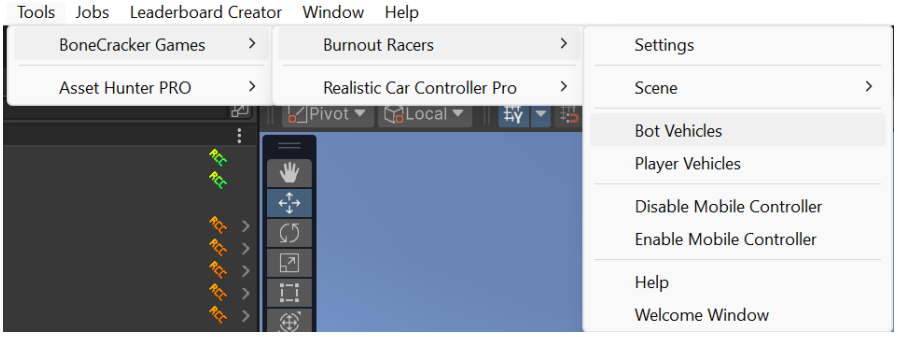
All player vehicles can be accessed from the **Tools** menu. You can add, remove, or edit any player vehicle and its price. Creating and adding a new vehicle will be explained.

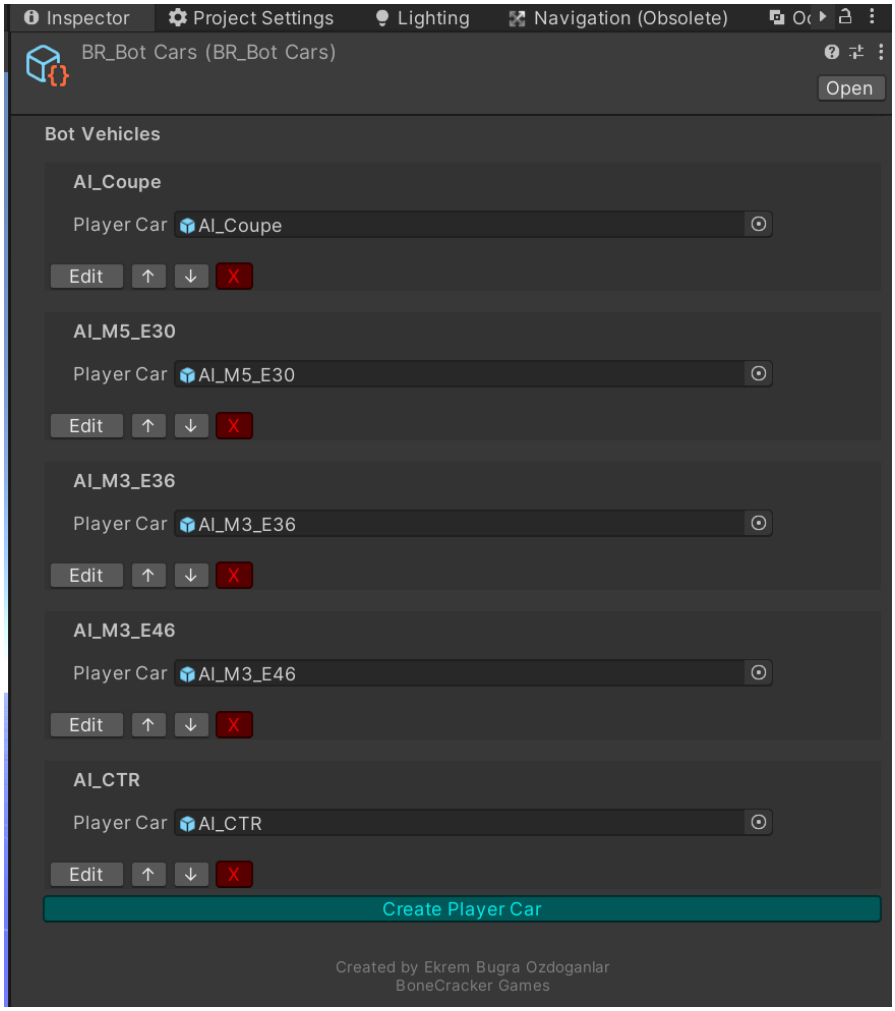




Bot Vehicles

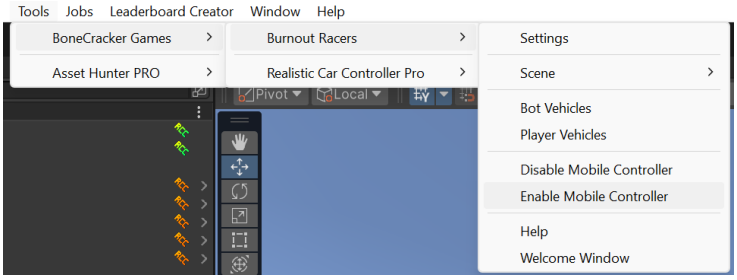
All AI bot vehicles can be accessed from the **Tools** menu. You can add, remove, or edit any AI bot vehicle. Creating and adding an AI car will be explained.





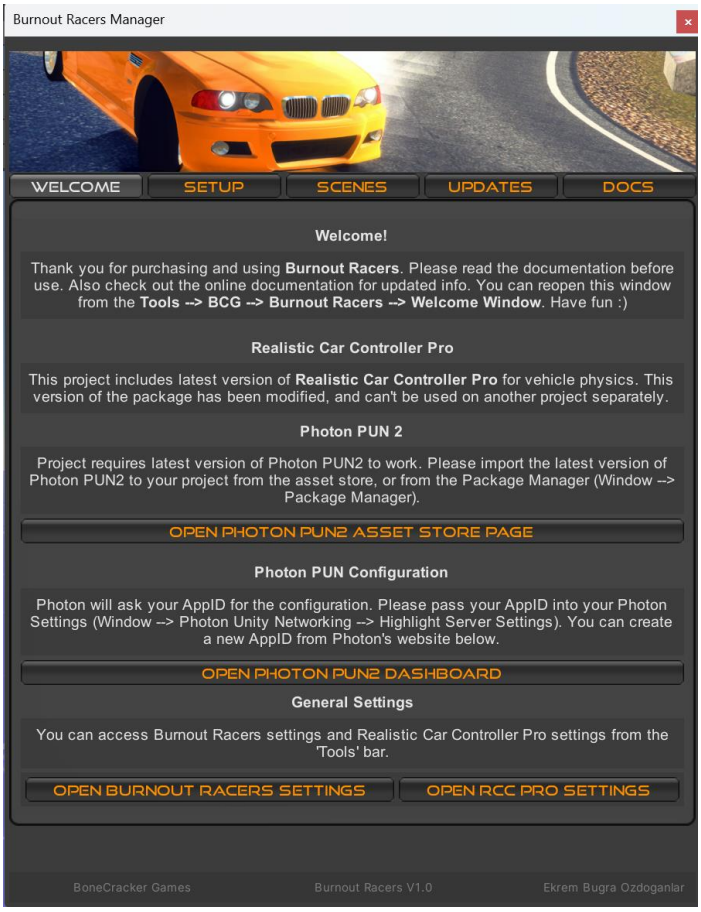
Enable / Disable Mobile Controllers

Mobile controllers can be enabled and disabled from the **Tools** menu. Please read the "**Performance on Mobile Devices**".



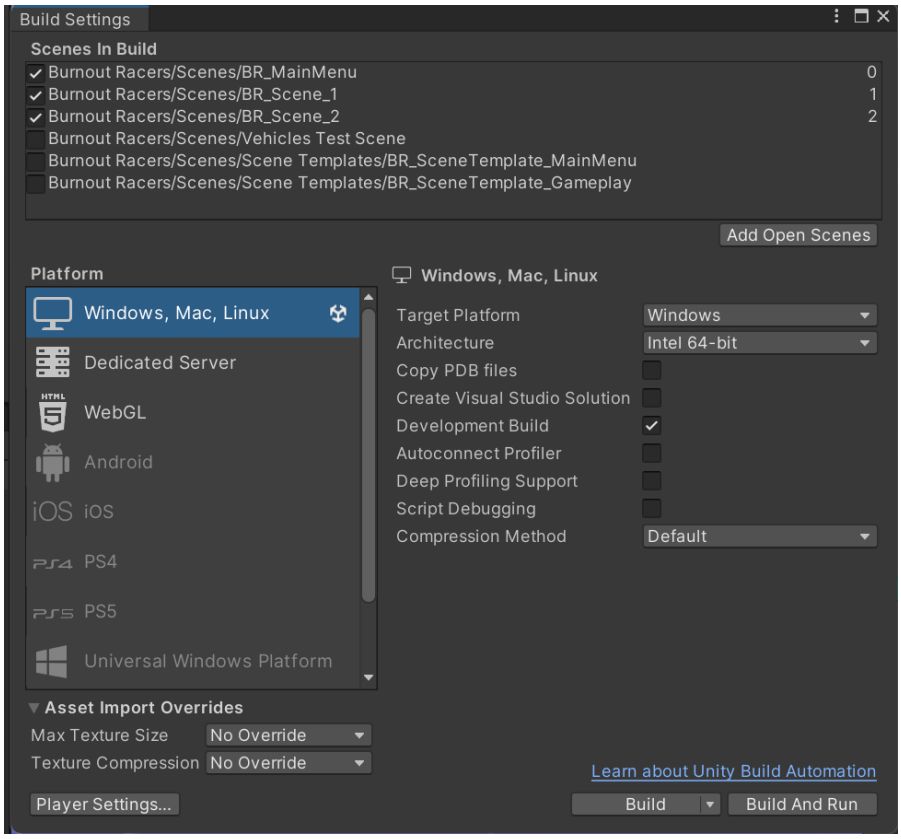
Welcome Window

Welcome window will appear automatically when you import and install the project for the first time. You can get access to it again from the tool menu.



How to Add New Scenes?

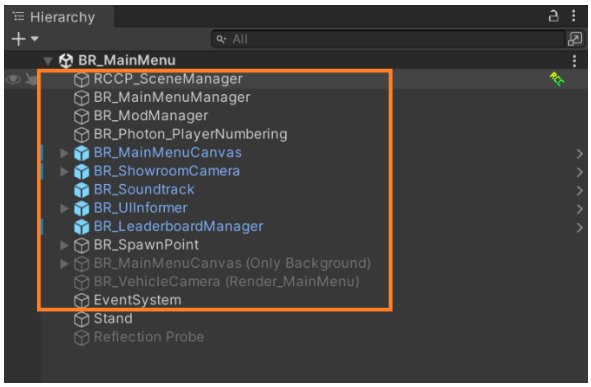
Project includes one main menu and two gameplay scenes. You can check the demo scenes in the **Build Settings** (**File → Build Settings**).

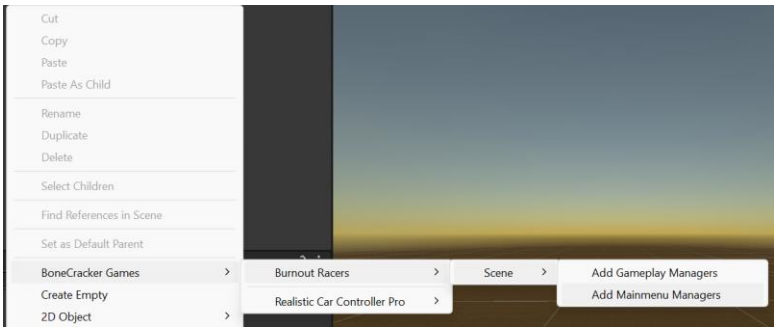


When the game starts, first scene will be loaded as main menu scene. Player's scene for gameplay selection action will be saved and game will load the selected level. Main menu and gameplay scenes have different systems. Therefore, you need to use different managers. To understand how the system works, you can check out the demo scene setups of main menu and gameplay.

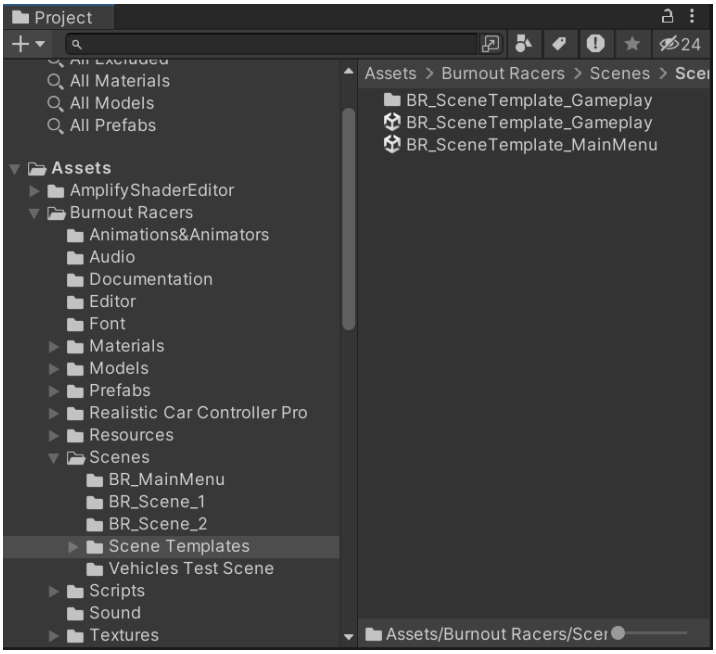
Adding a Main Menu Scene

Main menu requires these managers in the scene. These managers can be created directly from the BR's tool menu.



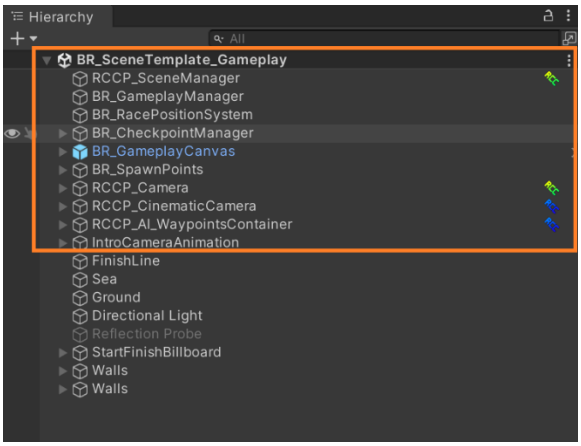


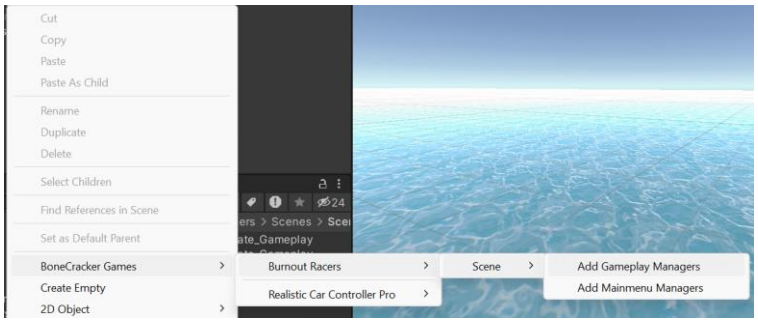
By using the "**Add Mainmenu Managers**" all managers will be created and configurated automatically. Also, you can use the template scenes as well which can be found in the scene setups folder. These templates include only managers, nothing else.



Adding a Gameplay Scene

Gameplay scenes require these managers in the scene. These managers can be created directly from the BR's tool menu.





How the Scenes Work?

Main menu and gameplay scenes have different managers.

Main Menu Workflow

Gameplay scenes have separated sections such as BR, RCCP, Lighting, Environment, and Others. Let's start with Burnout Racers managers. Lighting category has the lights, and environment category has the scene objects.

BR_MainMenuManager is managing all events generally. Spawns the player vehicle, switches them, selects / purchases them. Handles UI panels, sliders, and texts. All UI buttons will be using this manager's public methods. This is a singleton class; you can get access to it by using instance. Example; *BR_MainMenuManager.Instance*

BR_ModHandler is managing all events related to the modification. Once the player interacts with any kind of modification system, this script will be used to handle the UI, showroom camera, and some other systems. It doesn't do the modification directly on the vehicle, only managing the UI panels along with the showroom camera angles.

BR_Photon_PlayerNumbering is assigning unique numbers to each online player. This works only if player is in a room. Everyone in the room will have unique numbers, so they can spawn at different locations. This class has don't destroy on load; this means that the gameobject will not be destroyed while loading a new scene.

BR_ShowroomCamera is an orbit camera system used on the garage menu. Players can rotate the camera with inputs. This camera is enabled and active all the time. The other camera system is **BR_VehicleCamera**. This camera is used to render texture of the selected player vehicle. And thix texture will be displayed on the main menu UI canvas. This camera is enabled and active only when the main menu is enabled.

BR_Soundtrack is soundtrack system used on multiple scenes. This class has don't destroy on load; this means that the gameobject will not be destroyed while loading a new scene. Soundtracks can be changed from the **BR_Settings**.

BR_UIInformer is UI informer panel that displays important messages. You can also use it by accessing to the instance of the class.

BR_LeaderboardManager is an integration script for **Leaderboard Creator** asset. You need to import the asset and configurate the setup to use it. More info can be found in the "**Burnout Racers Leaderboard Integration**" documentation.

BR_SpawnPoint is used by **BR_MainMenuManager** to spawn the player vehicles. This is the location of the spawn point.

BR_MainMenuCanvas (Only Background) is used to display a background image at the garage menu. This gameobject will be enabled only on garage menu.

BR_VehicleCamera (Render_MainMenu) is used to render the player vehicle. This render texture will be used on the main menu to display the player vehicle at left side of the screen.

All stated scripts have been commented detailly, you may want to check them out.

Gameplay Workflow

BR_GameplayManager is managing all events generally. Manages overall gameplay. Spawns player / AI vehicles, observes them, countdown for start, starts the race, pause / resume, finishes the race, etc... It doesn't handle the UI section, this will be managed by **BR_UICanvasManager**. **BR_GameplayManager** is the main and base system for all of them. Everything has been commented detailly in the script.

BR_UICanvasManager is managing all UI gameobjects in the scene. All UI buttons, dropdowns, toggles will be using this manager's methods. Script will get access to the other system if necessary, through the **BR_GameplayManager**.

BR_WaypointsContainer is containing the race path. AI racers will be using this path. Also, this path will be used by the **BR_RacePositionManager** to calculate the race positions. It's important to have the accurate waypoint path in the scene, otherwise AI bots and race positioning system will have some problems.

BR_SpawnPoints is containing the spawn points for the players and AI bots. It has eight different spawn locations. Maximum spawn count of the vehicles is eight. It's been selected by the **BR_GameplayManager** in the scene.

BR_IntroCameraAnimation is containing a fixed animation clip with a camera inside. When the animation is triggered, it will be positioned correctly to render the player vehicle before the countdown starts. This has been selected and managed by **BR_GameplayManager** in the scene. This is optional, you don't have to use the intro camera if you don't want to. Just remove the intro camera gameobject from the scene, intro section will be bypassed.

BR_RaceFinisher is basically a trigger collider to finish the race. It will detect the passed vehicles on the finish line. This gameobject has an empty script, but it will be used by **BR_PlayerManager**. If vehicle can finish the race at that moment (if vehicle's lap is final lap), race will be ended for that vehicle.

Above managers and systems are part of the **Burnout Racers**. Let's have a look other managers and systems such as the vehicle controller (**Realistic Car Controller Pro**). Detailed information about the vehicle controller can be found in the RCCP's documentation folder.

RCCP_SceneManager

Scene manager that contains current player vehicle, current player camera, current player UI, current player character, recording/playing mechanism, and other AI vehicles as well. You can get access to this manager by *RCCP_SceneManager.Instance*. More info can be found it he RCCP's documentation.

RCCP_Camera

Camera controller for the player vehicle. Includes seven different camera modes with many customizable settings. It doesn't use different cameras on your scene. Simply it parents the camera to their positions that's all. No need to be Einstein. But we don't need to use other camera modes in this project. Only TPS, FPS, and fixed mode will be used. I don't think using the cinematic camera mode or top camera mode would be useful for race games. However, you can still enable and use them if you wish. More info can be found it he RCCP's documentation.

RCCP_FixedCamera

RCCP Camera's mode. It will be only enabled and used when the local player finishes the race. More info can be found it he RCCP's documentation.

Everything has been commented detailly in the scripts, you may want to check them out. I would recommend you start with **BR_GameplayManager** script, which is the main base manager of the scene.

This is how the main menu and gameplay scenes work. Of course we didn't go into the details. But I'm sure you'll have more useful information in the scripts. As I said before, I've explained everything in the scripts to understand how it works. My goal is to bring you a project that you can change and reskin it, not release the same game without touching anything. You're meant to be reskinning the project with your own content. That's why you could find comments and explanations in every script.

Events

Burnout Racers has these events which you can use in your own scripts by listening to them.

BR_GameplayManager.OnLocalPlayerSpawned (BR_PlayerManager Player)

BR_GameplayManager.OnLocalRaceStarted

BR_GameplayManager.OnLocalCountdownStarted

BR_GameplayManager.OnLocalRaceFinished (BR_PlayerManager player)

BR_GameplayManager.OnLocalRacePaused (bool state)

Example;

```
private void OnEnable(){
BR_GameplayManager.OnLocalPlayerSpawned += GameplayManager_OnPlayerSpawned;
}

private void OnDisable(){
BR_GameplayManager.OnLocalPlayerSpawned -= GameplayManager_OnPlayerSpawned;
}

 /// <summary>
 /// When the player spawns (local player).
 /// </summary>
private void GameplayManager_OnPlayerSpawned(BR_PlayerManager Player) {
   // Your code here when the player spawns.
}
```

Photon RPC Events

Burnout Racers is using RPC events to keep things synchronized across the network players. These events are listed below.

KickPlayerRPC = Used to kick the target player in the main menu.

NetworkPlayerSpawned = Used to add the spawned player to the list.

PlayerJoinedAndReady = Used to check all players before starting the race.

StartRaceRPC = Used to start the race on all clients at the same time.

SetRacePositionRPC = Used to set position number of each player.

StartCountdownRPC = Used to start the countdown on all clients at the same time.

RacerFinishedRPC = Used to add the finished player to the list.

How to Create and Add New Player Vehicles

All vehicles in the project have been powered by Realistic Car Controller Pro. You need to create your own new vehicles according to the RCCP's guidelines. Once you create your vehicle, you must create a prefab version of it, and select it in **BR_PlayerCars** asset.

1. Create a new vehicle according to RCCP's guidelines.
2. Create a prefab version of it. Usually, all player vehicles are located in Resources/Player Cars folder. Simply click the "**Create Prefab**" button on your inspector panel to do this. This button is in **BR_PlayerManager** component.
3. Select the prefab in **BR_PlayerCars** asset (*Tools → BCG → BR → Player Vehicles*).

Tutorial video of creating a new vehicle from scratch can be found below.

https://youtu.be/A_gKISKpEns?si=oTW8-sThunT12Wdu

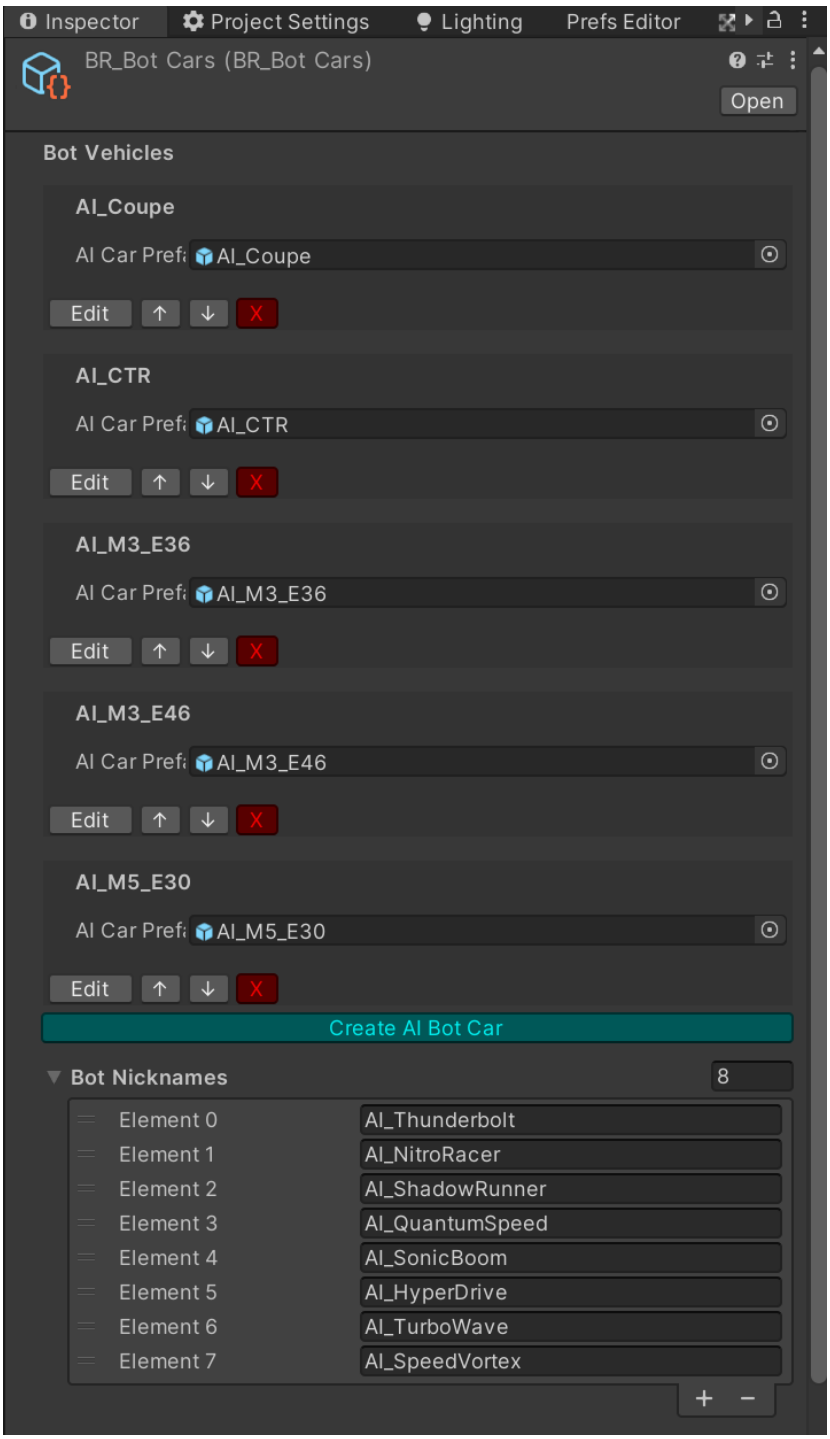
How to Create and Add New AI Vehicles

AI vehicles are just like the player vehicles. Only difference is, AI component of the vehicle is taking control of it. You need to create your own AI vehicles according to the RCCP's guidelines. Once you create your vehicle, you must create a prefab version of it, and select it in **BR_BotCars** asset.

1. Create a new vehicle according to RCCP's guidelines.
2. Add "AI" component to the vehicle as an addon component. You'll need to add "Other Addons" addon component to the vehicle in order to use AI.
3. Create a prefab version of it. Usually, all AI vehicles are located in Resources/AI Cars folder. Don't click the "Create Prefab" button on your inspector panel to do this.

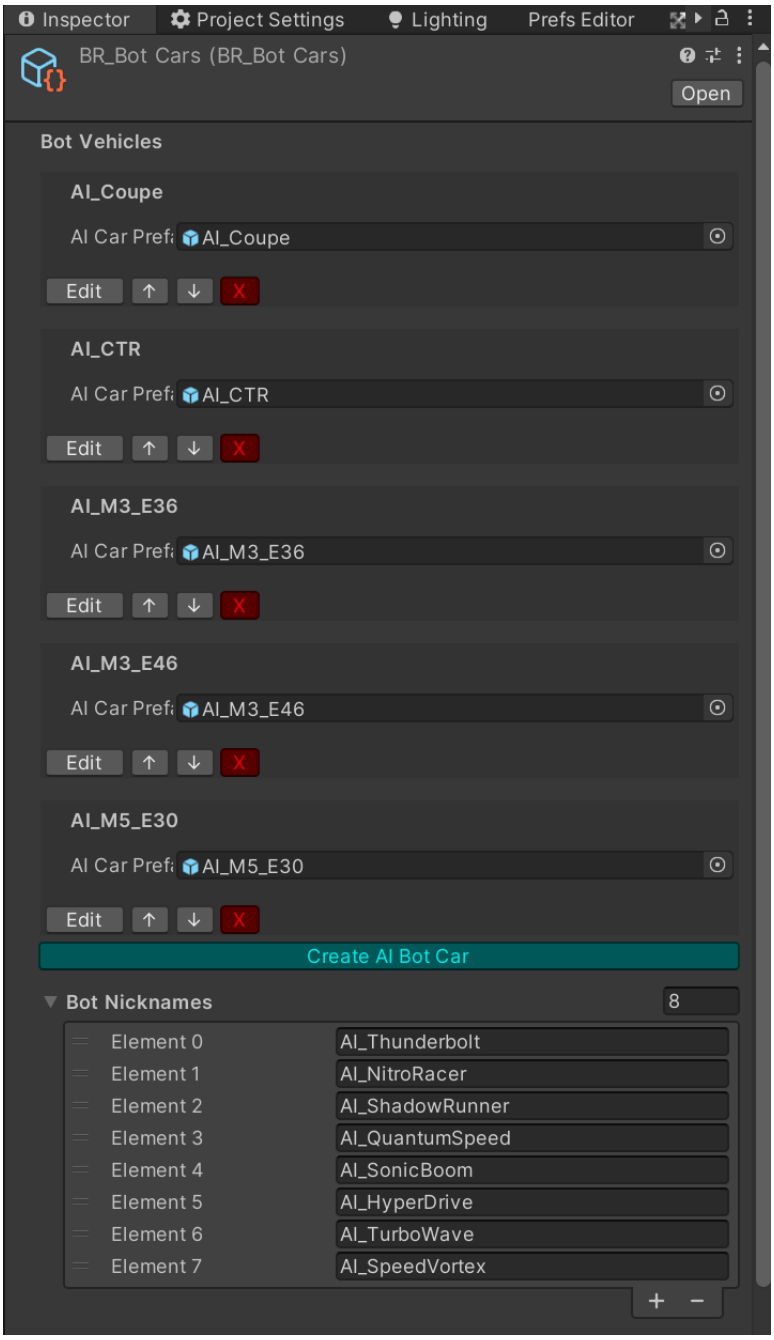
Simply drag and drop your vehicle gameobject to the Resources/AI Cars folder. This will create a prefab version of it.

4. Select the prefab in **BR_BotCars** asset (***Tools → BCG → BR → AI Bot Vehicles***).



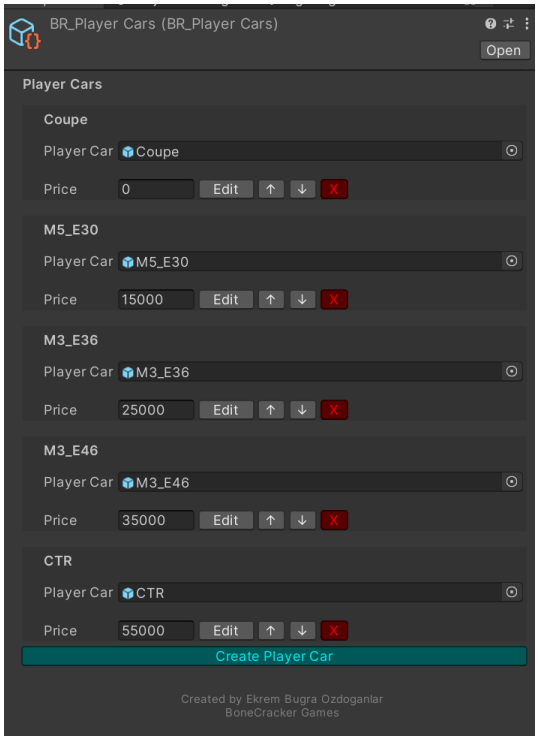
How to Change AI Names

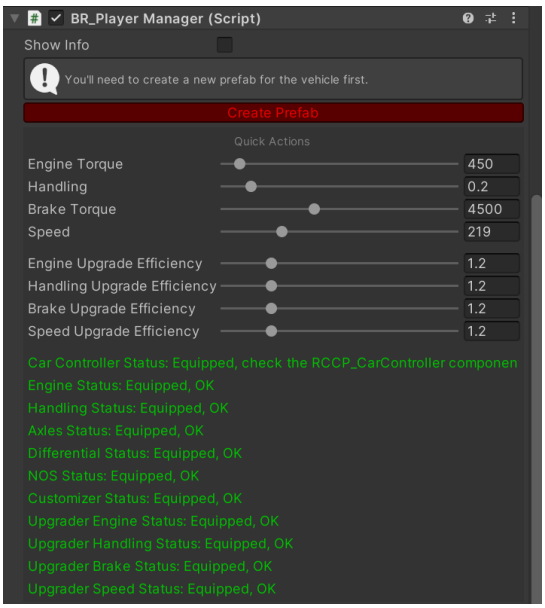
You can get access to the nickname list of AI vehicles from **Tools → BCG → BR → AI Bot** Vehicles. Manager will use these nicknames on spawned AI vehicles in the scene.



How to Change Player Vehicle Stats and Price

You can get access to the player vehicles list from **Tools → BCG → BR → Player Vehicles**. All selectable player vehicles are listed in this scriptable object. You can edit their prices and unlocked states. You can also change engine torque, handling, brake torque, maximum speed of the vehicle quickly from **BR_PlayerManager** component attached to the vehicle.



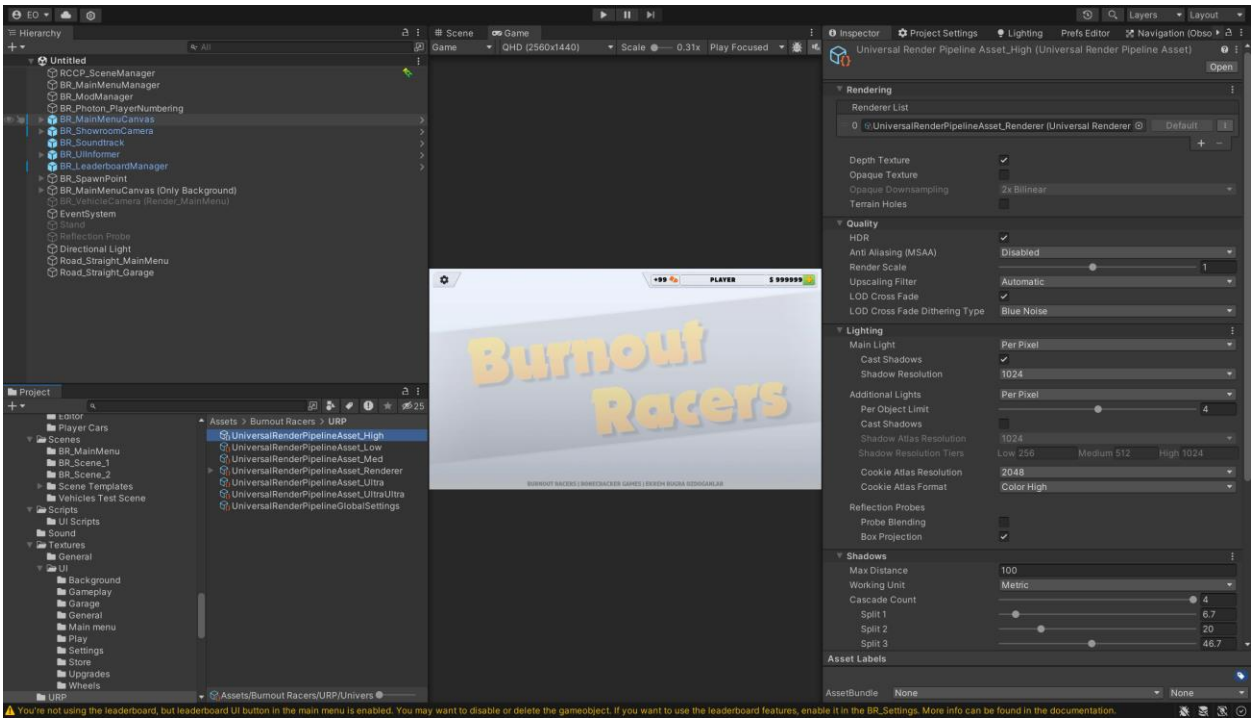


Performance on Mobile Devices (Shadows, and Post Processing)

Realtime Shadows

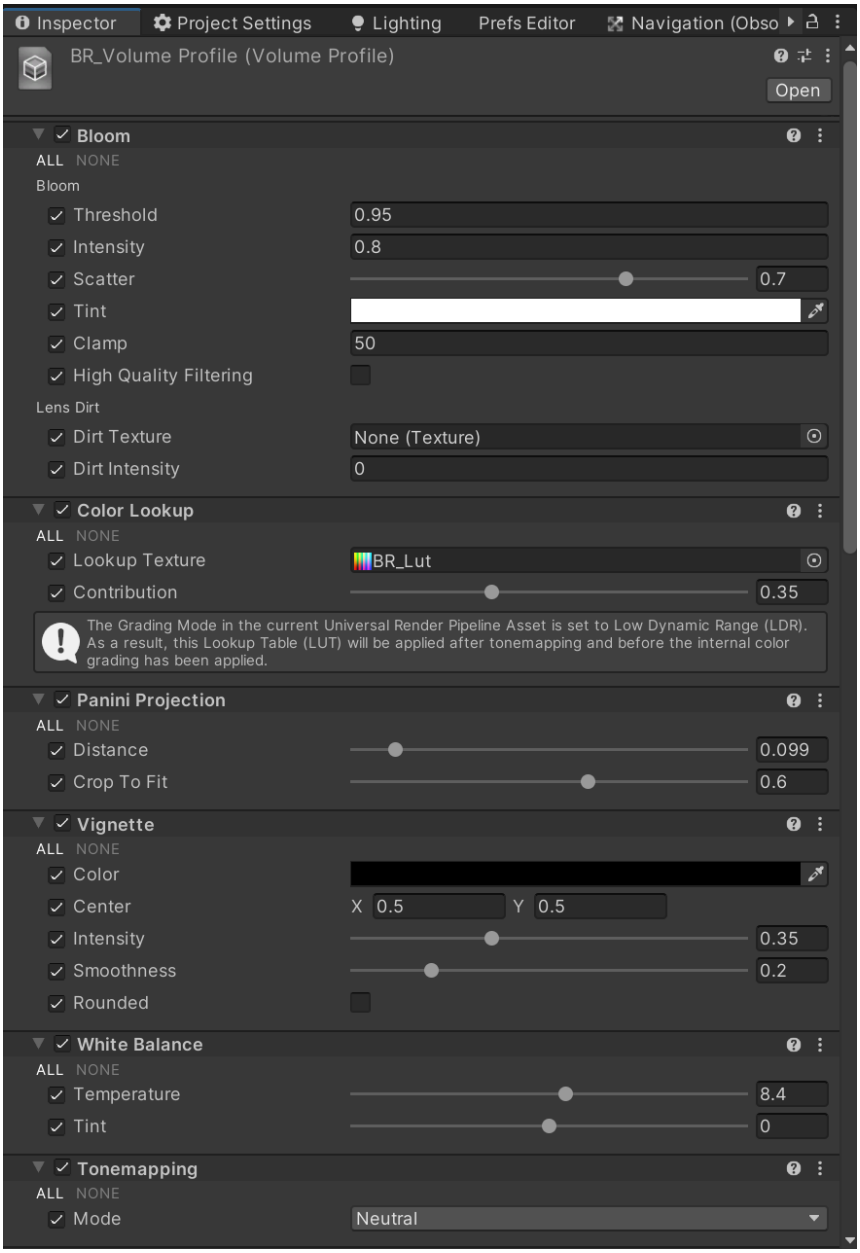
It’s recommended that you should be disabling the realtime shadows on the mobile devices. Consider using baked lightmaps, because realtime shadows are still too much expensive for mobile devices. You can disable the realtime shadows or lower shadow resolution from your URP asset.

Each quality level is using different URP preset. You can find all URP preset assets in the URP folder. Low, medium, high, ultra, and ultraultra. Ultraultra is used to take quality screenshots and videos inside the game. Not recommended to use that. Simply select each urp preset and disable the realtime shadows. You may still want to use them, if this is the case, you can lower the resolution for better performance.



Post Processing and Profiles

Project is using post processing effects as default. It's recommended to use it, because URP lens flares won't work without it. However, all cameras in the game have volume profile named "**BR_VolumeProfile**". This profile includes some expensive methods such as bloom, color correction, lut, and tonemapper. You may want to disable them for better performance on mobile devices. "**BR_VolumeProfile**" can be found in the URP folder. You can select he profile in your project and disable or remove the methods inside. This profile has been used by all cameras in the game.



Contact & Support

Please include your invoice number while sending me an email. I'll be responding within a day. I may not respond on the weekends.

Developed and published by BoneCracker Games

Ekrem Bugra Ozdoganlar

BoneCrackerGames@gmail.com