This document will guide you on how to use this asset to make your own cool racing games in Unity3D.

For any questions, suggestions or complaints feel free to contact me at ian.izzy94@gmail.com and I will respond promptly.

Let’s get started!

V 1.0.5

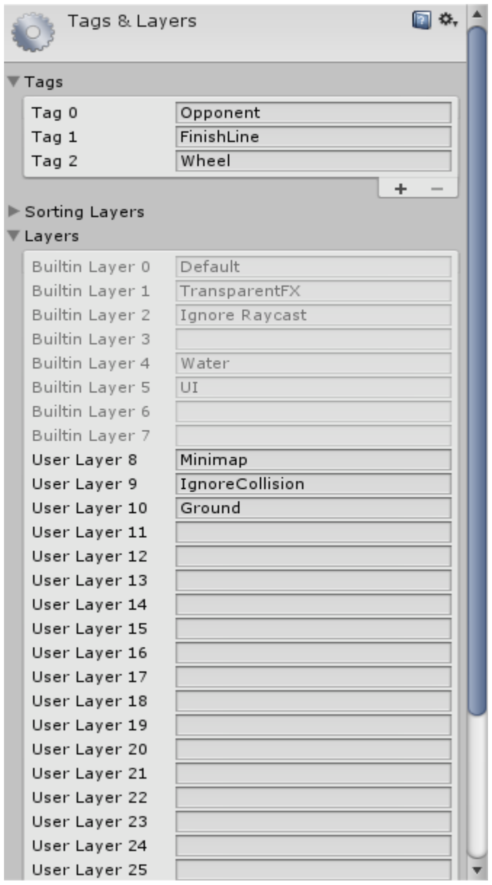
Content

Getting Started

Project Settings

Tags & Layers

You may need to add tags & layers to your project.
Go to Edit/Project Settings/Tags & Layers and add the following :
Tags > "**Opponent**", "**FinishLine**" & "**Wheel**"
Layers > "**Minimap**", "**IgnoreCollision**" & "**Ground**"



Tags

Tag 0	Opponent
Tag 1	FinishLine
Tag 2	Wheel

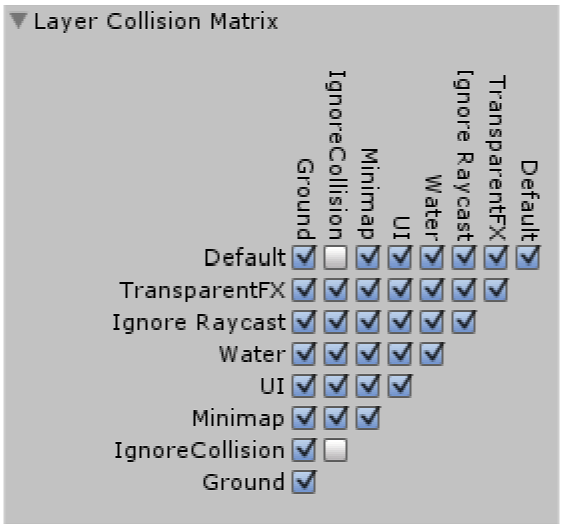
Sorting Layers

Layers

Builtin Layer 0	Default
Builtin Layer 1	TransparentFX
Builtin Layer 2	Ignore Raycast
Builtin Layer 3	
Builtin Layer 4	Water
Builtin Layer 5	UI
Builtin Layer 6	
Builtin Layer 7	
User Layer 8	Minimap
User Layer 9	IgnoreCollision
User Layer 10	Ground
User Layer 11	
User Layer 12	
User Layer 13	
User Layer 14	
User Layer 15	
User Layer 16	
User Layer 17	
User Layer 18	
User Layer 19	
User Layer 20	
User Layer 21	
User Layer 22	
User Layer 23	
User Layer 24	
User Layer 25	

Physics Settings

You may also need to tweak the Layer collision matrix.
Go to Edit/Project Settings/Physics and make the following changes :



	Ground	IgnoreCollision	Minimap	UI	Water	Ignore Raycast	TransparentFX	Default
Default	✓	☐	✓	✓	✓	✓	✓	✓
TransparentFX	✓	✓	✓	✓	✓	✓	✓	
Ignore Raycast	✓	✓	✓	✓	✓	✓		
Water	✓	✓	✓	✓	✓			
UI	✓	✓	✓	✓				
Minimap	✓	✓	✓					
IgnoreCollision	✓	☐						
Ground	✓							

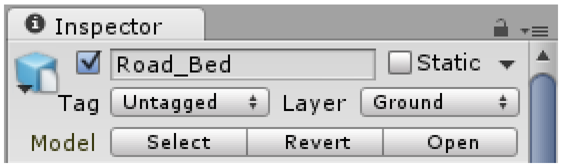
This is to ensure that collisions with objects in the “IgnoreCollision” layer are ignored by every other object other than the ground.

Part 1. Race environment setup

The race environment is the entire track set up – the path, the triggers, the spawnpoints, the minimap and the race manager.

1. Set up the Path

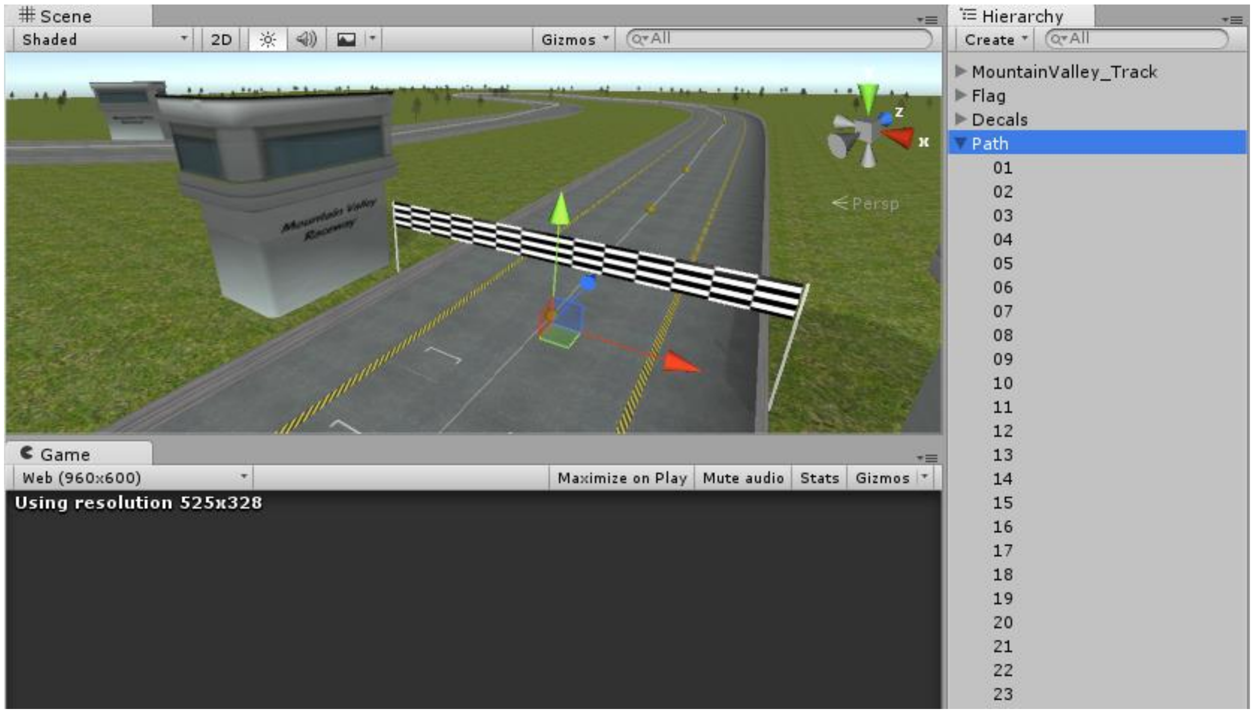
Firstly, select your race track(or it's collision mesh) and put it in the "**Ground**" layer as follows :



The next thing to do is set up a path around your track. A path consists of multiple nodes. Nodes are used as waypoints in the race.

To set up a path, go to **Window/RacingGameStarterKit/Path Creator**.

This will create a PathCreator object in your scene. Position the path at your track's starting point then begin to create a path around your track by duplicating the child node as follows :



Keep doing so until you have a **complete** path around your track. Nodes don't necessarily have to be close to each other. Ensure the **last node** is close to the finish.

After you have a complete path around your track, click on the "**Finish**" button on your path creator object's inspector. This will create a "WaypointCircuit" component.

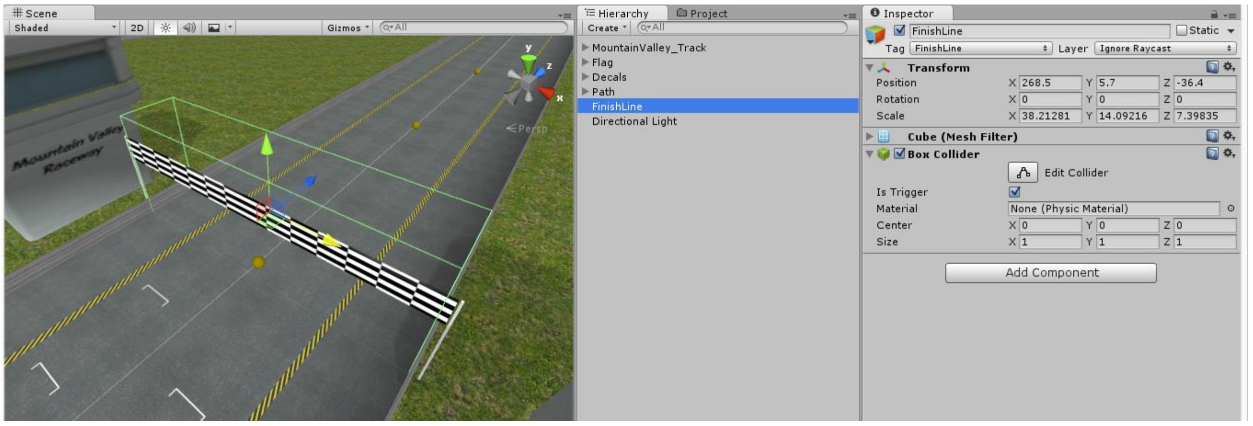
Lastly, on the WaypointCircuit component, click "**Assign using all child objects**" button.

2. Set up the Triggers

There are 2 types of triggers that are used – **Finish Line** & **Checkpoint**

Finish line

To set up a finish line trigger, create a box collider trigger and give it the "**FinishLine**" tag and put it in the "**Ignore Raycast**" layer :



```
FinishLine                          Static
Tag  FinishLine        Layer  Ignore Raycast
Transform
Position     X 268.5      Y 5.7        Z -36.4
Rotation     X 0          Y 0          Z 0
Scale        X 38.21281   Y 14.09216   Z 7.39835
Cube (Mesh Filter)
Box Collider
             Edit Collider
Is Trigger   ✓
Material     None (Physic Material)
Center       X 0          Y 0          Z 0
Size         X 1          Y 1          Z 1
```

Place this trigger at your race track's finish line.

Checkpoints

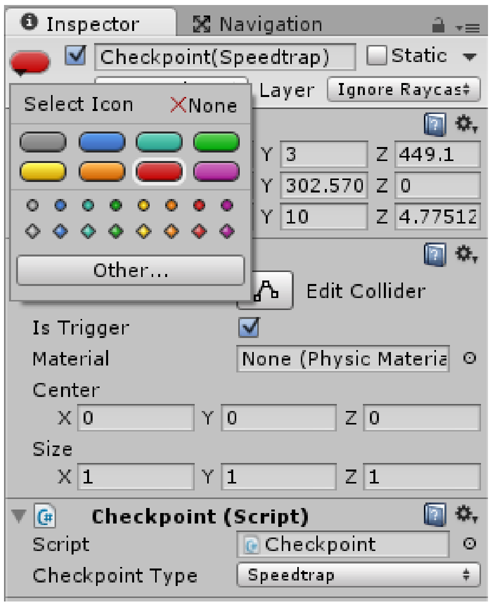
There is currently 1 type of checkpoint :

Speed Trap - used in the “Speed Trap” race type to capture a racers speed.

To set up checkpoints for your race track, go to **Window/RacingGameStarterKit/Checkpoint Creator**.

Afterwards, place your checkpoints strategically around your track.

You can give your Checkpoints gizmo icons to make them easier to manage :



Checkpoint(Speedtrap) ☐ Static
Layer Ignore Raycast

Y	3	Z	449.1
Y	302.570	Z	0
Y	10	Z	4.77512

Is Trigger ✓
Material None (Physic Materia

Center

X	0	Y	0	Z	0

Size

X	1	Y	1	Z	1

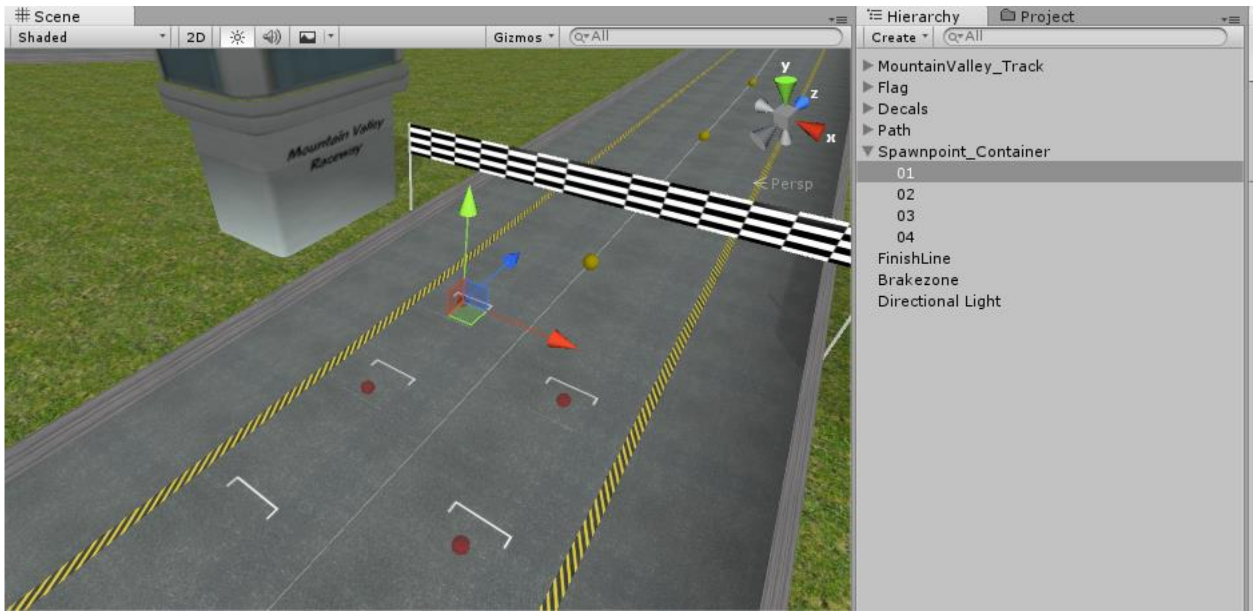
Checkpoint (Script)
Script Checkpoint
Checkpoint Type Speedtrap

3. Set up the Spawnpoints

Spawnpoints define where the racers will initially spawn.

To create spawnpoints, go to **Window/RacingGameStarterKit/Spawnpoint Creator**.

This will create a new SpawnpointContainer. Position the spawnpoint child objects to where you want the racers to spawn :



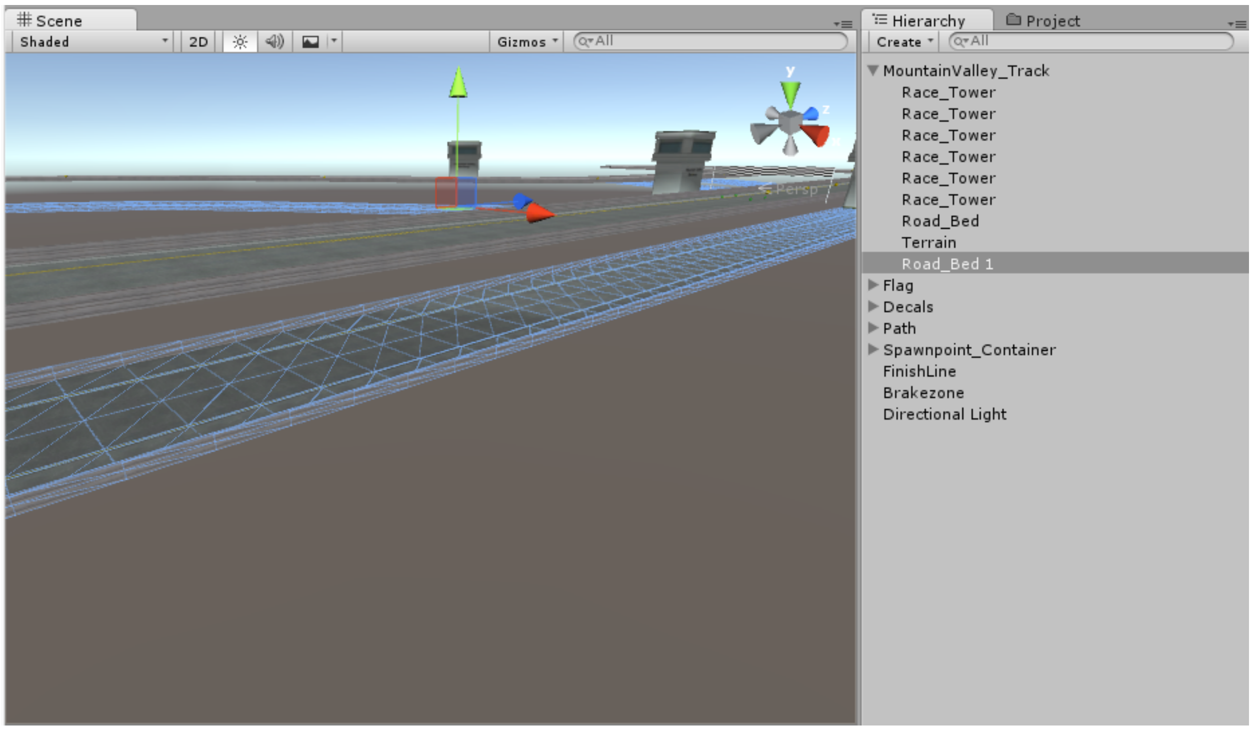
Keep in mind that racers will spawn facing in the spawnpoint’s direction so ensure the spawnpoint’s rotation is how you want it.

Racers will spawn according to the spawnpoint hierarchy arrangement. For example, in the picture above, “01” will be considered as position 1, “02” will be considered as position 2 and so on.

4. Set up the Minimap (Optional)

To set up a minimap :

- Duplicate your track and position it directly below your original track :



MountainValley_Track
  Race_Tower
  Race_Tower
  Race_Tower
  Race_Tower
  Race_Tower
  Race_Tower
  Road_Bed
  Terrain
  Road_Bed 1
Flag
Decals
Path
Spawnpoint_Container
FinishLine
Brakezone
Directional Light

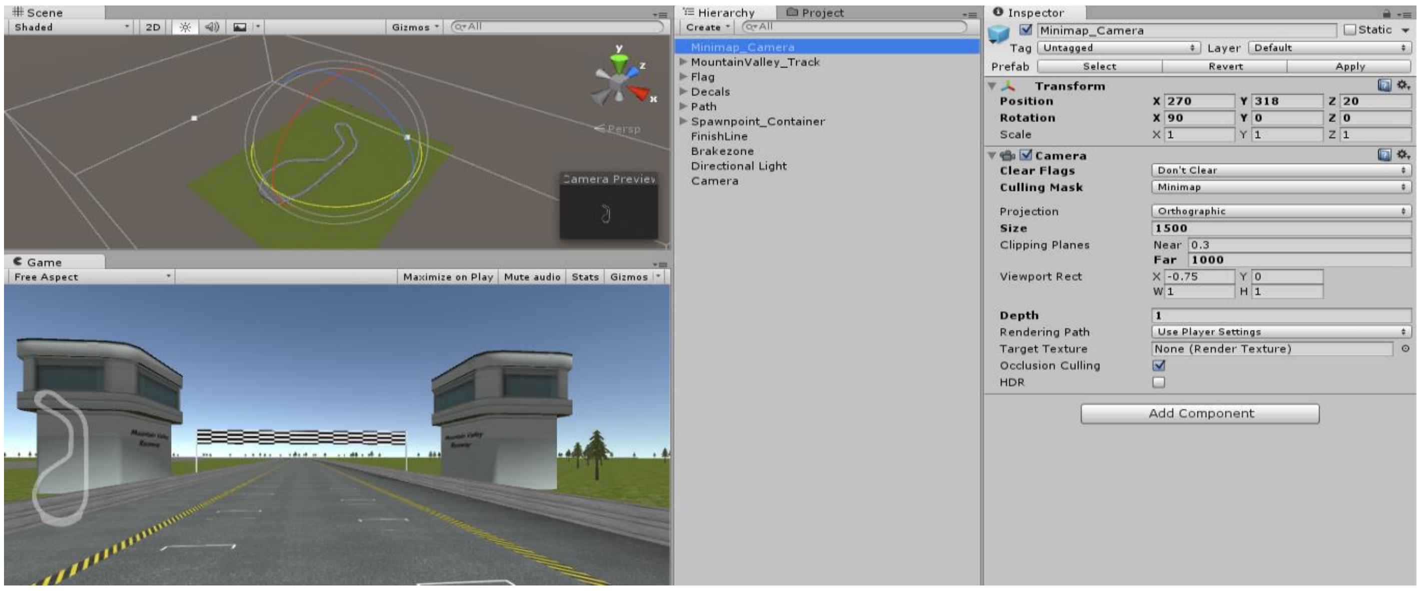
- Assign the “Minimap” layer to the duplicate track.

- Drag the MinimapCamera prefab from the Prefabs/Misc folder to your scene.

- Set the camera to “Don’t clear” flags, Orthographic and the Culling Mask to “Minimap” only.

- Position the camera using the transform component & Viewport rects to where you want it

- Give your duplicate track different materials, preferably the minimap material used in the demo scene and you should have something that looks like this now :



Minimap_Camera
MountainValley_Track
Flag
Decals
Path
Spawnpoint_Container
FinishLine
Brakezone
Directional Light
Camera

```
Minimap_Camera                              Static
Tag  Untagged          Layer  Default
Prefab  Select   Revert   Apply
Transform
Position          X 270    Y 318    Z 20
Rotation          X 90     Y 0      Z 0
Scale             X 1      Y 1      Z 1
Camera
Clear Flags       Don't Clear
Culling Mask      Minimap
Projection        Orthographic
Size              1500
Clipping Planes   Near 0.3
                  Far  1000
Viewport Rect     X -0.75  Y 0
                  W 1      H 1
Depth             1
Rendering Path    Use Player Settings
Target Texture    None (Render Texture)
Occlusion Culling ✓
HDR
```

5. Set up the Waypoint arrow(Optional)

To set up a waypoint arrow, you need to make sure that there is an active gameObject named “WaypointArrow” in your scene. Place this arrow as a child of your camera.

You can also set up a waypoint arrow by following these steps :

- Drag the PlayerCamera prefab from Prefabs/Misc folder.
- Simply enable or disable the child “WaypointArrow” before playing if you want to use it or not.

Note : Ensure your car has the WaypointArrow.cs attached. This is done automatically when configuring cars from v1.0.4

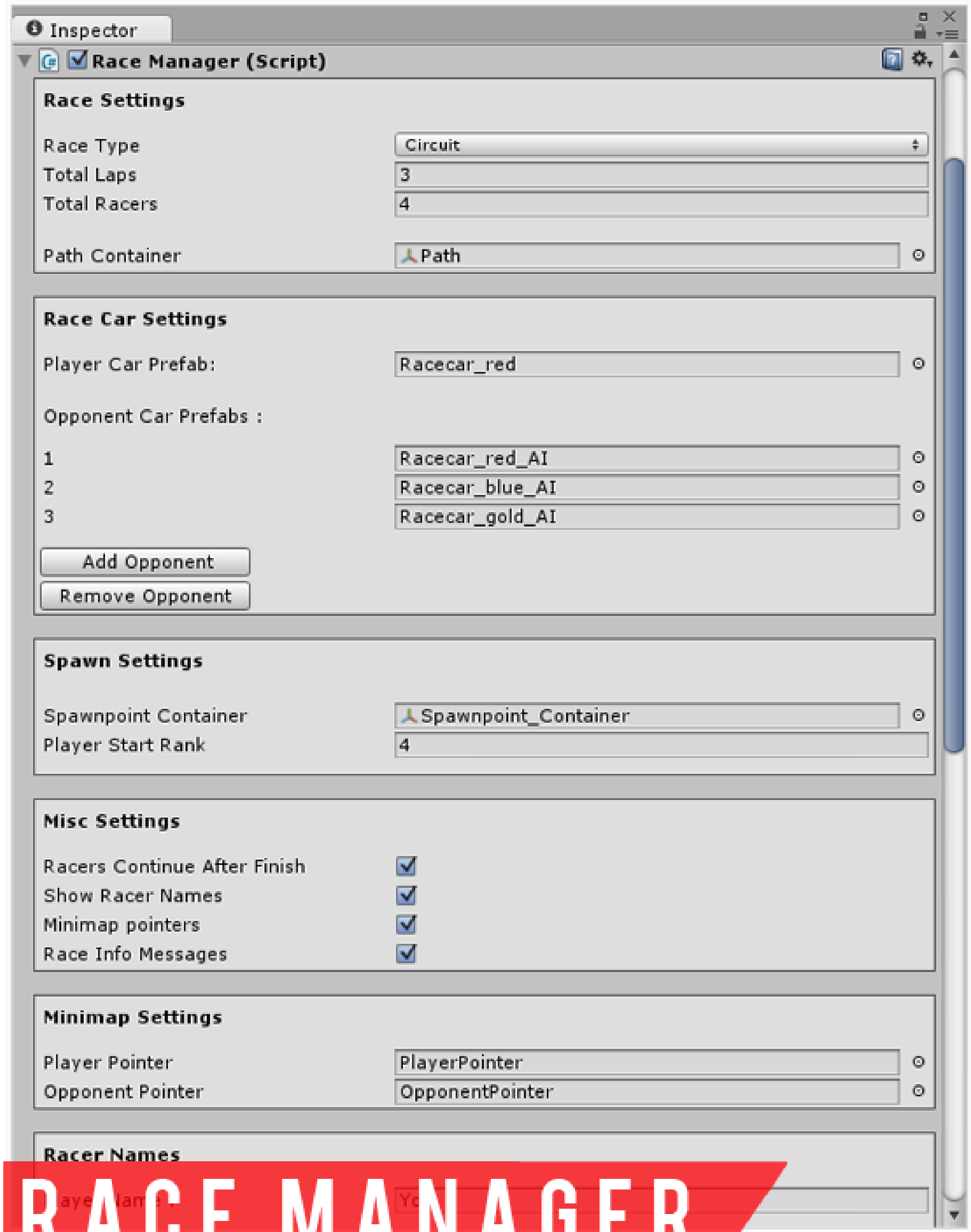
6. Set up the Race Manager

The race manager handles race logic.

To create a Race Manager, go to **Window/RacingGameStarterKit/Create Race Manager**.

This will create a race manager in your scene. The race manager contains other important components as well : RankManager, RaceUI & SoundManager.

In the race manager you can set the race type, laps, number of opponents, racer vehicles , names, minimap pointers and more :



Inspector

Race Manager (Script)

Race Settings

Race Type	Circuit
Total Laps	3
Total Racers	4
Path Container	Path

Race Car Settings

Player Car Prefab:	Racecar_red

Opponent Car Prefabs :

1	Racecar_red_AI
2	Racecar_blue_AI
3	Racecar_gold_AI

Add Opponent

Remove Opponent

Spawn Settings

Spawnpoint Container	Spawnpoint_Container
Player Start Rank	4

Misc Settings

Racers Continue After Finish	✓
Show Racer Names	✓
Minimap pointers	✓
Race Info Messages	✓

Minimap Settings

Player Pointer	PlayerPointer
Opponent Pointer	OpponentPointer

Racer Names

RACE MANAGER

Race Settings

- **Race type**.
  -**Circuit Race** is a normal race where racers have to complete a certain number of laps.

  -**Lap Knockout** is a last man standing type race. Every lap the racer in last place is knocked out.

  -**Time Trial** is a solo race where you aim to set a track best time. You MUST set the "**StartPoint**" object to determine where the car will begin driving from. You can also set a Ghost vehicle in this mode by toggling "**Use Ghost Vehicle**". The Ghost vehicle will follow the path you took allowing you to race against yourself and set the best possible track time.

  -**Speed Trap** is a race whereby the racer with the highest total speed wins. Speed is captured using Speed trap Checkpoints.

- **Total laps** is self-explanatory. On a lap knockout race, the total laps are set to an appropriate number depending on the number of racers.

- **Total racers** are the total number of racers that will race(player included). Make sure your spawnpoints match the this value.

Race Container Settings

- **Path container** is the gameObject that contains the path. The race manager will automatically prompt you to create or assign one if null.

- **Spawnpoint container** is the gameObject that contains the spawnpoints. The race manager will automatically prompt you to create or assign one if null.

- **Checkpoint container** is the gameObject that contains the checkpoints. The race manager will automatically prompt you to create or assign one if null.

Race Car Settings

- **Player Car Prefab** is your car.

- **Opponent Car Prefabs** are the opponent cars. You can set as many as you want. The race manager will spawn them at random.

Spawn Settings

- **Player Start Rank** is the position you will start as.

Misc Settings

- **Racers continue after finish** Should all the racers keep driving after finishing the race?

- **Show Racer Names** Should opponent racer names appear on top of them? Make sure you set a racer name prefab under "**Racer Names"** if set to true. The racer name prefab Must contain a "RacerName.cs" attached to it.

- **Show Minimap Pointers** Should there be pointers on top of the racers? Make sure you set the values under "**Minimap Settings"** if set to true. The minimap pointers Must contain a "RacerPointer.cs" attached to it.

- **Race Info Messages** will show useful race messages such as, "Final Lap", "New best time" , "Speed trap" & knocked out racer and more texts if set to true. Make sure you set a "RaceInfo" text in RaceUI for this to work.

- **Race countdown delay** is how long to wait(in seconds) before starting the countdown.

Racer Names

- **Player Name** is your name. If left empty, it will load a string from the PlayerPref "PlayerName".

- **Opponent Names** are generated from a .txt file located in the Resources folder. To change or add the names, open the "RacerNames.txt" in the resources folder .

Rank Manager

The rank manager handles setting car ranks/positions according to their race completion.

Race UI

Race UI handles displaying all the UI in the scene. UI is displayed using UGUI. You will need to create a Canvas and place your UI elements within it. Check out the UI tutorials if you are new to this.

Make sure you assign all the variables to avoid getting errors. Most of the variables are self-explanatory but I will go through a few :

- **Race Info** is a text used for displaying race info messages. To display a message from any of your scripts, simply call :

  "RaceUI.instance.StartCoroutine(RaceUI.instance.ShowRaceInfo("My race info text",3.0f));"

  This will display that text for 3 seconds on screen.

- **Race Standings** is an array of texts that are used to display at racer ranks during and at the end of a race. Each element has an **InRaceStanding**(optional) with 1 text and an **EndRaceStanding** with 3 texts – Position of the racer, Name of the racer & Time of the racer. It is important to arrange the elements in order e.g Position 1 should be element[0], position 2 should be element[1] etc.

- **Speedometer**(*Optional*) **:**
  - **Needle** is the speedometer needle.
  - **Min Needle Angle** is the angle the needle should be when the car is not moving
  - **Max Needle Angle** is the angle the needle should be when the car is moving at top speed.
  - **Rotation Multiplier** is the amount of rotation added to the needle. This is used for precision.

- **Screen Fade**(Optional) **:**
  - **ScreenFade** is the fade Image.
    Important Note : Should be the last child of the Canvas(i.e not a child of any panel). Also, assign a "CanvasGroup" component to it and uncheck all checkboxes.

  - **FadeSpeed** is the rate at which fade occurs.

- **FadeOnStart/Exit** determines whether the screen should fade when the scene loads and when it is exited.

You can always tweak the RaceUI.cs script by opening it and making changes that will suit your racing game.

Sound Manager

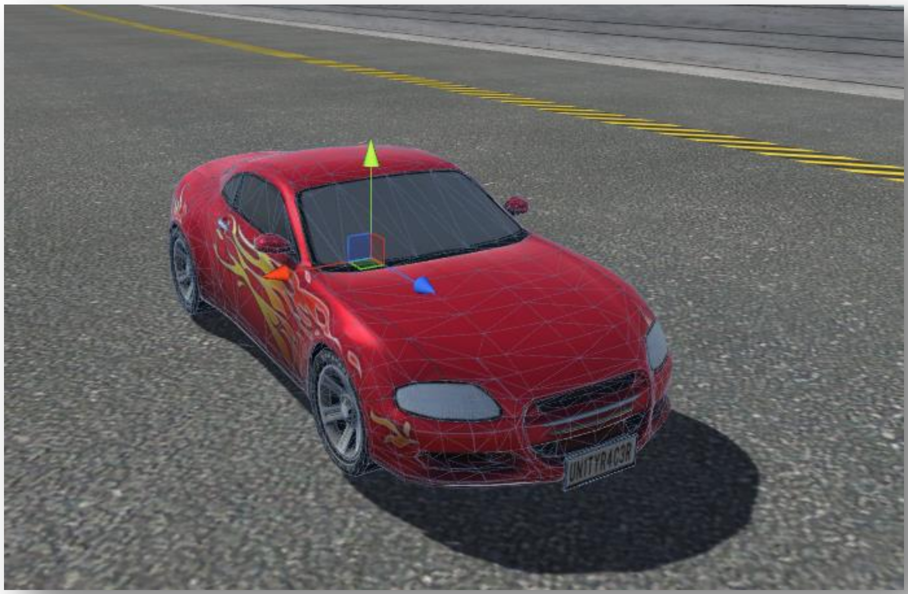
SoundManager handles playing sounds in the race and also playing background music. The sounds its assigned to play by default are the countdown and go sounds when the race starts and also car crash sounds. Using the SoundManager is simple, just add a new game sound in the list. Whenever you want to play the sound, call any of the methods in the script.

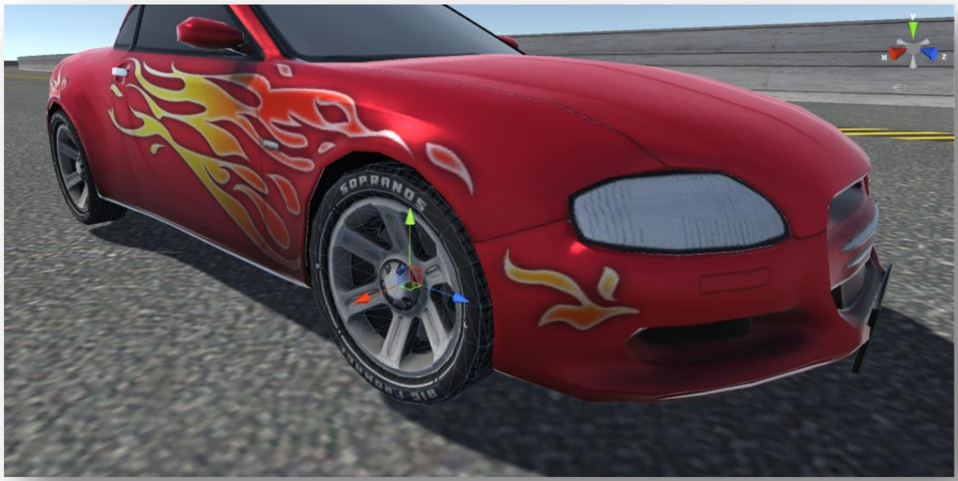
The Sound Manager also handles playing music. Assign a music track and set the volume and you will have looping background music.

Part 2. Race car setup

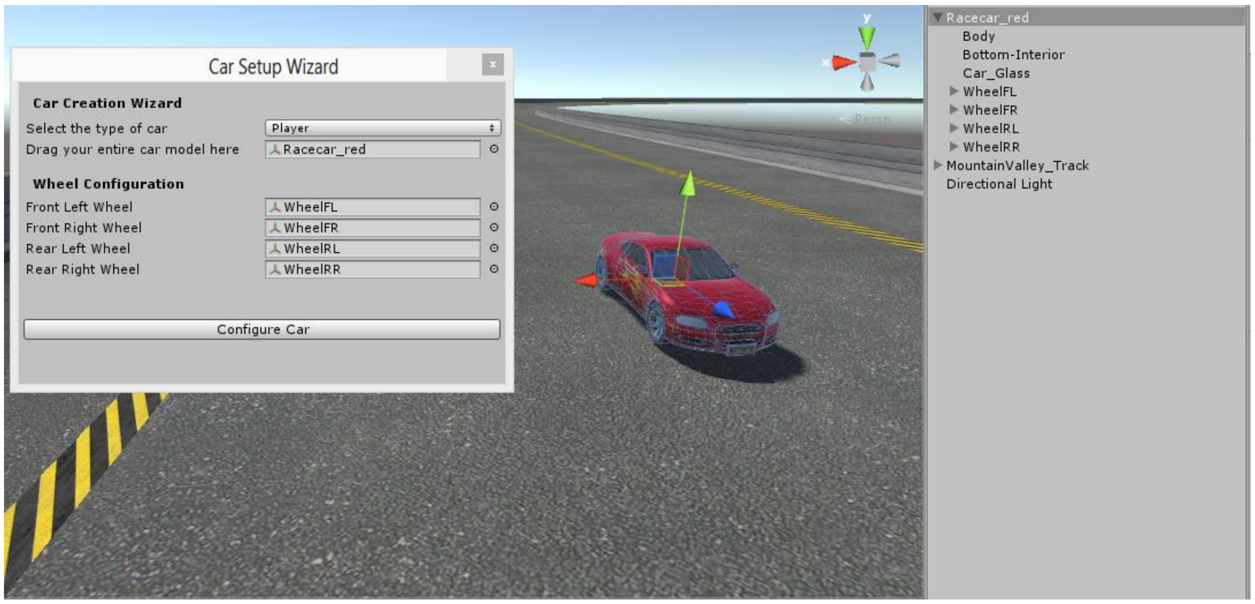
1. Car set up

Setting up cars has been made extremely easy in this kit by the use of the “Car Setup Wizard”. Before configuring your car, please make sure your car axis and wheel axis are correct. Z should point forward, X should point right and Y should point up as follows:





Next, go to **Window / RacingGameStarterKit / Car Setup Wizard**, fill in the details and configure the car when ready :


Car Setup Wizard

Car Creation Wizard

Select the type of car: Player

Drag your entire car model here: Racecar_red

Wheel Configuration

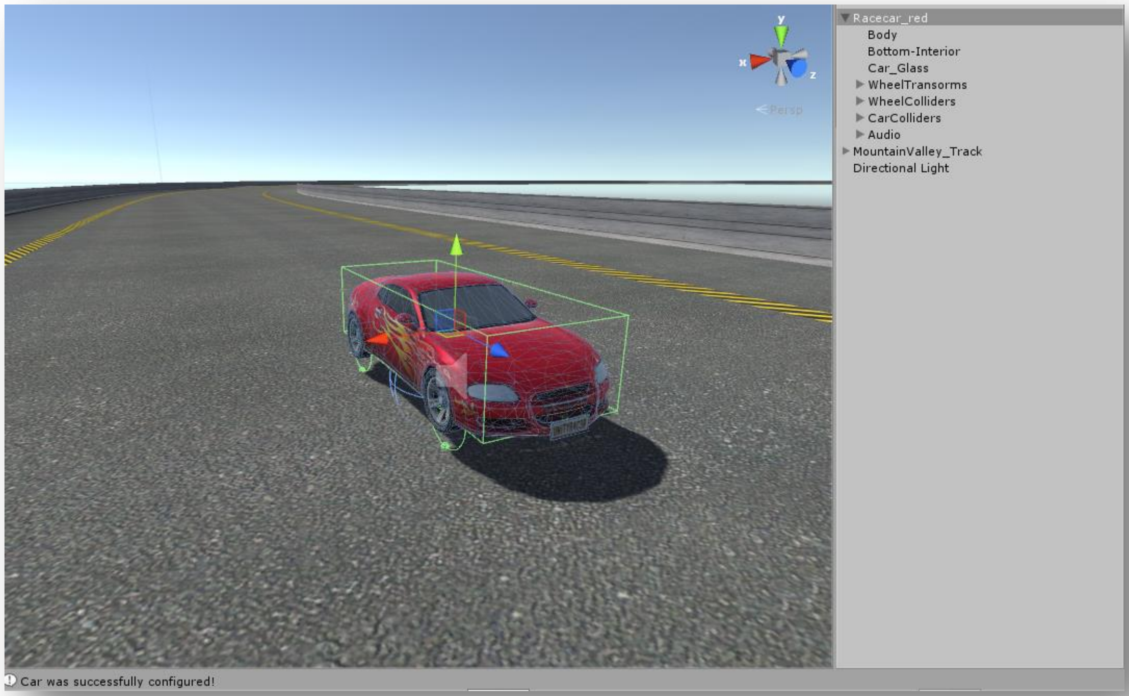
Front Left Wheel: WheelFL

Front Right Wheel: WheelFR

Rear Left Wheel: WheelRL

Rear Right Wheel: WheelRR

After configuring the car, your car will have all the necessary components to race. Some components may need tweaking such as the car collider and the WheelColliders :



Car was successfully configured!

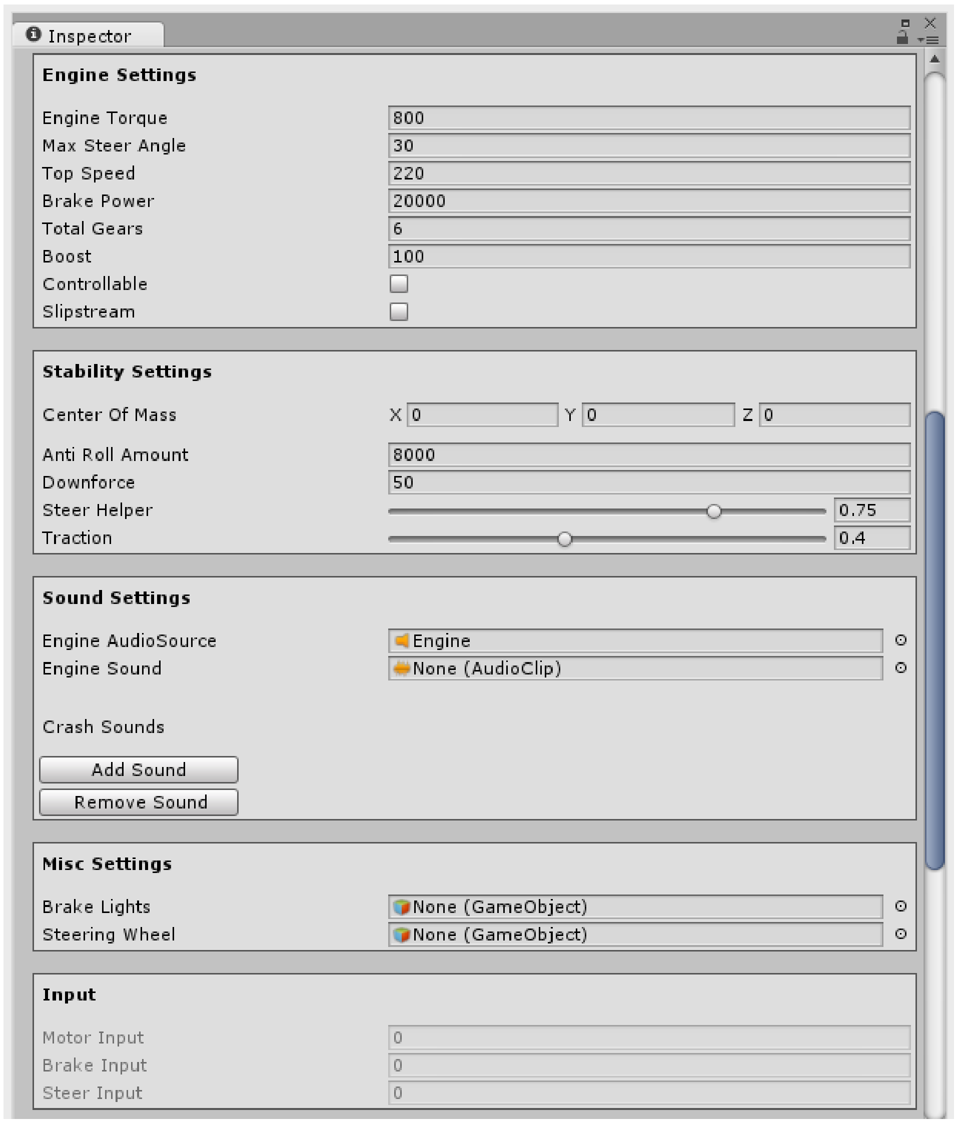
At this point, you can set “controllable” to true on the Car_Controller component and test drive your car with the RaceManager gameObject **disabled**.

To set up an **AI car**, open the Car Creation Wizard and select “Opponent” instead of “Player”.

“Tip : You can also set up an AI car directly from your player car by changing the tag to “Opponent”, replacing the “PlayerControl.cs” with “OpponentControl.cs” and removing the “CameraSwitcher.cs” & “WaypointArrow.cs” component.

2. Car configurations

Your cars will have a Car_Controller component attached. This defines the car's properties. Tweak the settings to your liking :



Engine Settings

Engine Torque	800
Max Steer Angle	30
Top Speed	220
Brake Power	20000
Total Gears	6
Boost	100
Controllable	☐
Slipstream	☐

Stability Settings

Center Of Mass	X 0 Y 0 Z 0
Anti Roll Amount	8000
Downforce	50
Steer Helper	0.75
Traction	0.4

Sound Settings

Engine AudioSource	Engine
Engine Sound	None (AudioClip)

Crash Sounds

Misc Settings

Brake Lights	None (GameObject)
Steering Wheel	None (GameObject)

Input

Motor Input	0
Brake Input	0
Steer Input	0

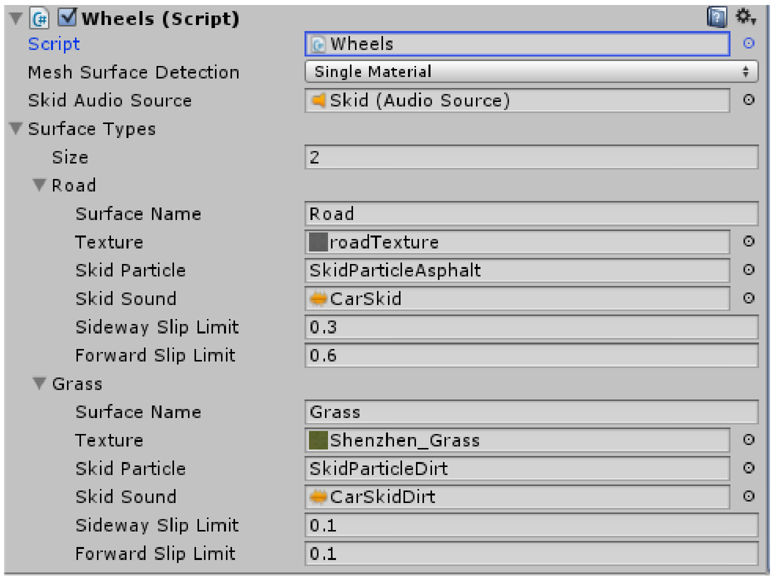
Most of these values will be pre assigned on by the “Car Setup Wizard” to make things easier for you. Most of these values are also self-explanatory but I will go through a few :

- **Boost** is the amount of forward force added to the car as you accelerate.
- **Slipstream** determines whether the car will be able to slipstream other cars. If set to true, the car will get a speed boost when driving behind other cars.
- **Downforce** is the amount of downward force added to the car as it speeds up.
- **Steer Helper** is the amount of added turn force when steering.
- **Traction** is the amount of grip given to the powering wheels

<u>Tips for making your cars faster</u>

- Increase each wheel colliders forward friction value to 2.
- Increase the boost amount
- Reduce rigidbody mass
- Increase the engine torque (avoid setting a high value(> 2000)!)

Next, go to your car’s wheel colliders and configure the settings in the “Wheels.cs” component :



Wheels (Script)

Script	Wheels
Mesh Surface Detection	Single Material
Skid Audio Source	Skid (Audio Source)
Surface Types	
Size	2
Road	
Surface Name	Road
Texture	roadTexture
Skid Particle	SkidParticleAsphalt
Skid Sound	CarSkid
Sideway Slip Limit	0.3
Forward Slip Limit	0.6
Grass	
Surface Name	Grass
Texture	Shenzhen_Grass
Skid Particle	SkidParticleDirt
Skid Sound	CarSkidDirt
Sideway Slip Limit	0.1
Forward Slip Limit	0.1

The Wheels.cs component handles surface detection and generating particles and sounds depending on the texture wheel is on. Surface detection works on both Terrains & Renderers.

If your track mesh uses more than one texture, set the “MeshSurfaceDetection” to **MultipleMaterial**.

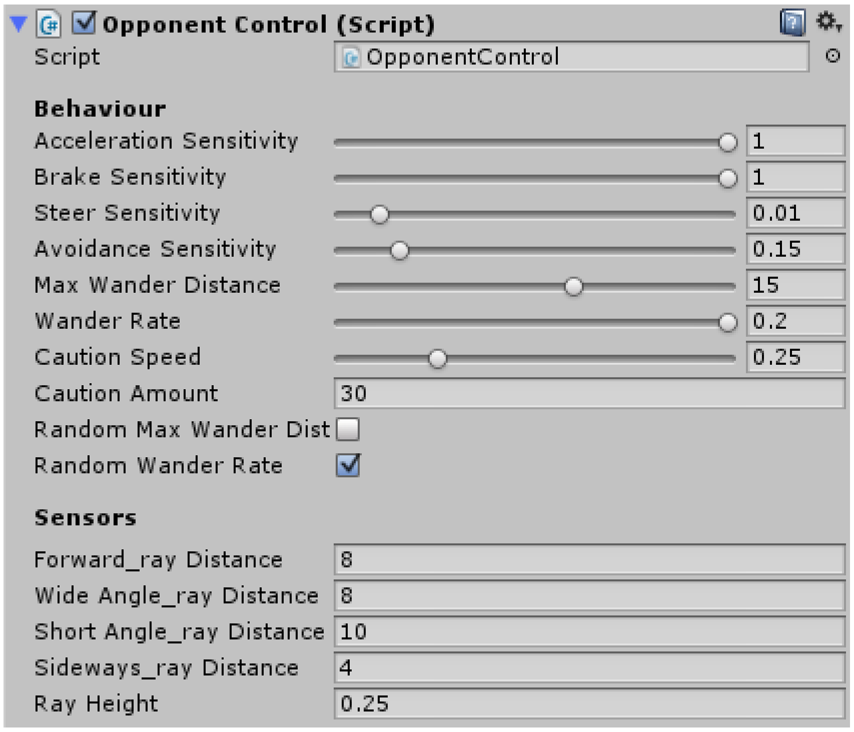
If your track mesh only uses one texture, set the “MeshSurfaceDetection” to **SingleMaterial**.

Note : Using MultipleMaterial mesh surface detection is performance intensive!

Each surface type has the following settings :

- **Texture** is the surface's texture
- **Skid Particle** is the particle to be emitted when in a drift
- **Skid Sound** is the sound played when drifting on this surface
- **Sideway/Forward Slip Limit** is the amount of wheel friction slip needed for the skid particle to emit.

AI Configuration



Opponent Control (Script)

Script	OpponentControl
Behaviour	
Acceleration Sensitivity	1
Brake Sensitivity	1
Steer Sensitivity	0.01
Avoidance Sensitivity	0.15
Max Wander Distance	15
Wander Rate	0.2
Caution Speed	0.25
Caution Amount	30
Random Max Wander Dist	☐
Random Wander Rate	✓
Sensors	
Forward_ray Distance	8
Wide Angle_ray Distance	8
Short Angle_ray Distance	10
Sideways_ray Distance	4
Ray Height	0.25

The OpponentControl.cs handles controlling the AI input. You can give each AI different behaviors by tweaking the behavior settings. I will go through what each one does :

- **Acceleration Sensitivity** is how sensitively this AI will use the accelerator.

- **Brake Sensitivity** is how sensitively this AI will use the brake.

- **Steer Sensitivity** is how sensitively this AI will steer to reach the next waypoint.

- **Avoidance Sensitivity** is how sensitively this AI will steer to avoid another car and/or obstacle. When tweaking this value consider checking the **Sensors settings** too.

- **Max Wander Distance** is how far the AI can travel(laterally) from the path. Don't set this value larger than your track's width

- **Wander Rate** is how often the AI will begin to wander

- **Caution Speed** is the percentage of the top speed that the AI will try to reach when it is fully cautious(e.g at a sharp corner)

- **Caution Amount** is how cautious the AI will be around corners.

- **Random Max Wander Distance** sets a random *MaxWanderDistance* on start.

- **Random Wander Rate** sets a random *WanderRate* on start.

Tip : play around with these values till you find the behavior that you want your AI to have.

Multiple Car Cameras

To set up multiple camera views for your car :

- Create 1 or more cameras as direct child objects of your car.
- Uncheck “Minimap” from your camera’s CullingMask
- Disable the AudioListener(to avoid warning messages on start).
- Press “V” to switch between cameras in play mode.

Check out the “CameraSwitcher.cs” for more insight about how camera switching is handled.

3. Mobile Controls

- Drag the “MobileUI” prefab from Prefabs/UI folder to your Canvas.

- Under your car’s PlayerControl.cs component, select “Mobile” control type.

The PlayerController.cs will find the UI buttons at runtime.

Finalizing the scene

Next, add the PlayerCamera prefab from Prefabs/Misc to your scene if you haven't already done so.

At this point, ensure that you have :

1. A fully configured race environment – Path, Spawnpoints, Triggers, Race Manager (with fully configured RaceUI variables).

2. Fully configured race cars (player and opponent cars).

   Lastly, add the player and opponent cars to the race manager's "Race Car Settings".

Part 3. Race Data

This part will give you insight of how to load selected cars , racer preferences & best times.

1. Loading Data

Cars & Preferences

If you open MenuScript.cs & DataLoader.cs you will understand how loading your selected cars and race preferences are handled.

Cars are loaded from the resources folder and assigned to the RaceManager on Awake().
Player name is loaded from a “PlayerName” PlayerPref string if not empty in the RaceManager.

Best Times

Your best time for each scene is stored in a string.
To access a best time simply load it by :
PlayerPrefs.GetString(“BestTime”+”YourSceneName”);

Congratulations!

You’ve set up a complete racing environment! Hit play and race!

If you encounter any problems or errors, please reference the demo scene or contact me at ian.izzy94@gmail.com

Thank you for your support!

Credits

The demo car ("Racecar") & the race music are from the Unity Car Tutorial Demo