T3Dmake

Traffic Racing complete kit

2017

Thank you for purchasing the Traffic Racing complete kit

Contents

Introduction

This kit was made to let you create traffic racing games easily. Add your own models and UI and play the game. Choose your car, buy it and choose one of the locations. Drive endless roads and avoid collisions with the randomly spawned cars.

About this project

While making a 3D runner game, I learned a lot about endless spawning and scrolling textures. For this project, I wanted to make a game that's more fun to play. So I made fast cars, 2 endless worlds, cars that prevent accidents and switch lines and a garage with some unlockable cars. There's lots of popular traffic racing games on the Google play store already which gave me inspiration. The hardest part of making an endless 3D racing game is to create the illusion of moving forward while actually having a static car. All cars seem to move forward, while they're moving backwards, the trees and houses aren't static, they move backwards even faster than the cars and even the distance isn't real distance, it's just calculated using car speed.

General system

To create the illusion of moving forward, everything needs to move the right way with the right speed. Otherwise it would turn out looking pretty weird. For me, one of the hardest things in this system was to have different cars with different high speeds. Because everything is moving except the car, I needed the car to control everything around it. All together, the system works as follows:

- *Car selected by player gets instantiated*
- *Car passes its high speed to the game manager*
- *The scrolltexture script finds the game manager*
- *Scrolltexture has a float as well, called 'scrollspeed'*
- *Scrollspeed uses the high speed value*

Then, all ground textures scroll using the scrollspeed of scrolltexture script. Houses, trees, bridges etc. move with scrollspeed * 6, to move as fast as the ground textures. Cars move the same direction as all other objects but slower, to make it look like they move forward and even the time between spawning new objects and cars is based on scrollspeed, to make sure the concentration of cars and objects doesn't depend on the car speed.

In the comments of for example, the 'EnvironmentSpawner' script, you'll see things like *'based on scrollspeed (car speed)',* that's because it than uses scrollspeed, while scrollspeed is most of the time the same as car speed.

Controls

Player car controls will automatically change based on the build settings. By default it uses keyboard controls. You can change these in the game manager (please change them in all racing scenes). If your platform is set to Android/IOS, players can choose between touch and accelerometer controls in the pause menu.

Resetting the game

Resetting the game (deleting PlayerPrefs to lock all cars and set coins to 0) is very easy. Just go to one of the racing scenes, go to the manager script and press delete PlayerPrefs data.

Garage

One of the main goals was to create an easily customizable garage. The garage is used to select and unlock new cars. Since coins and new cars are the main goal of the game, the garage is pretty important. The garage scene includes a start screen as well and that's also why the garage script handles the start panel. It's very easy to add your own cars to the garage and set a price for them. Also from the garage scene, you can open the race locations.

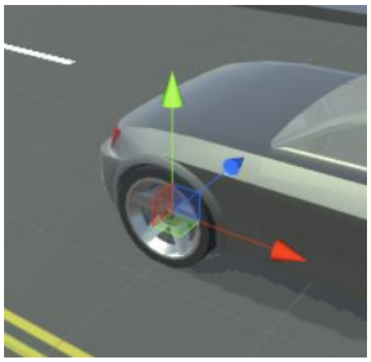
Creating and adding your cars

Using your own cars in the game is quite easy. There are a few steps to set up a car and use it in the garage.

- First, import your car model and drag it into the scene. Please make sure it's an empty with a mesh attached called 'car mesh'.
- Now, give your car one big box collider and a small one at the front.
- Set the small collider to 'Is Trigger'.
- Add the car controls script and a rigid body. Copy rigid body values from one of the drivable car prefabs.
- Then add the crash audio and turn off 'play on awake'.
- Copy smoke & boost particles of one of the drivable car prefabs and add them to your car (to the main empty object).

Then to add the wheels,

- Add your wheel model to the "car mesh" object and copy it 4 times.
- Position and rotate the wheels correctly.
- Add 4 empties to the car mesh as well (one for every wheel) and give each empty one of the wheel positions. Make sure you don't copy wheel rotations/scale.
- The empties should all be rotated the same, like this:

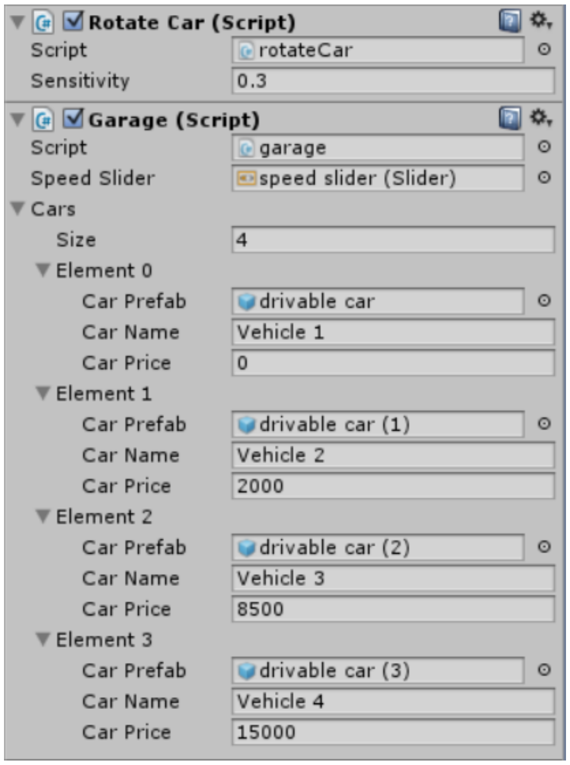


- Parent each wheel to its corresponding empty and add 'wheel' scripts to the empties. Please set rotation speed to around 100. Of course, only enable front wheel for the front wheels. For max steer rotation, you can use something like 25 and for rotation speed something like 30.

The car is done now. To let players unlock it,

- Go to the platform object (garage scene) and in the garage script, add a new car.
- Make a prefab of your car and add it to the garage script.
- Set a name and price for your car. The name is used to save your car with PlayerPrefs too.
- Finally, add your car prefab to the game manager of **each scene**.

Please note that the order of the cars in your garage and in the scenes, should be the same, otherwise the cars don't correspond.



Rotate Car (Script)

Script	rotateCar
Sensitivity	0.3

Garage (Script)

Script	garage
Speed Slider	speed slider (Slider)
Cars	
Size	4
Element 0	
Car Prefab	drivable car
Car Name	Vehicle 1
Car Price	0
Element 1	
Car Prefab	drivable car (1)
Car Name	Vehicle 2
Car Price	2000
Element 2	
Car Prefab	drivable car (2)
Car Name	Vehicle 3
Car Price	8500
Element 3	
Car Prefab	drivable car (3)
Car Name	Vehicle 4
Car Price	15000

Garage script containing the car list.
Attached to the platform object.

Viewing your cars

To let your players have a good impression of the car they are going to unlock and to show your cars, I added the possibility of rotating cars in the garage by dragging your mouse (or finger). This is done by writing a script, called ‘rotateCar’ and adding it to the platform. Every instantiated car gets instantiated as a child of the platform, so they rotate together with the platform.

Line switching cars

Line switching makes the game way more interesting and a bit more difficult as well. All vehicles switch lines and will automatically prevent accidents.

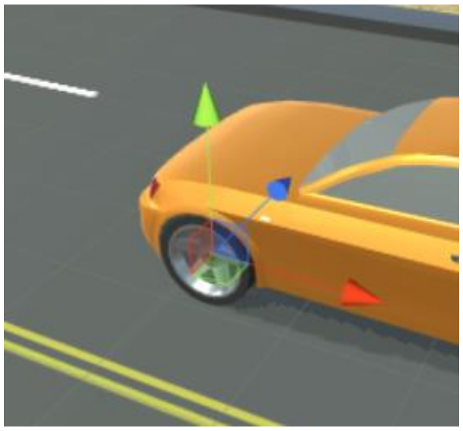
Setting up your own line switching cars

To set up your own line switching cars, there are a few simple steps.

- Import your car model (with separate wheels) and add it to the scene. Please make sure the main object is an empty with a mesh attached called “car mesh”. Give it (the main empty) a move object script and check car, change track and set a slow down distance. (Distance between car and the car in front of this car. I called it slow distance because it looks like the car slows down, while it actually moves faster…) If you have a normal size car, you should set this distance between 6 and 8, if you have a large vehicle like a bus, it should be around 10.
- Now, add a box collider and a rigid body and copy rigid body values of one of the ‘simple vehicles’ prefabs.
- Then, copy the left/right lights from one of the ‘simple vehicles’ prefabs and make sure you name them exactly the same. Please add these to the main empty object.

And finally please follow these steps to add the wheels:

- Add your wheel model to the “car mesh” object and copy it 4 times.
- Position and rotate the wheels correctly.
- Add 4 empties to the car mesh as well (one for every wheel) and give each empty one of the wheel positions. Make sure you don’t copy wheel rotations/scale.

- The empties should all be rotated the same, like this:

- Parent each wheel to its corresponding empty and add 'simple car wheel' scripts to the empties. Please set rotation speed to around 100.

After you've set up your car, add it to the object pool in both scenes

Preventing accidents

To prevent accidents between cars, the line switching cars check the road with rays, before moving. Also all moving objects with car checked, have a slow down distance. That distance is the length of the ray which checks for other cars. If another car hits the ray, the car moves faster to prevent accidents (in game it looks like the car slows down).

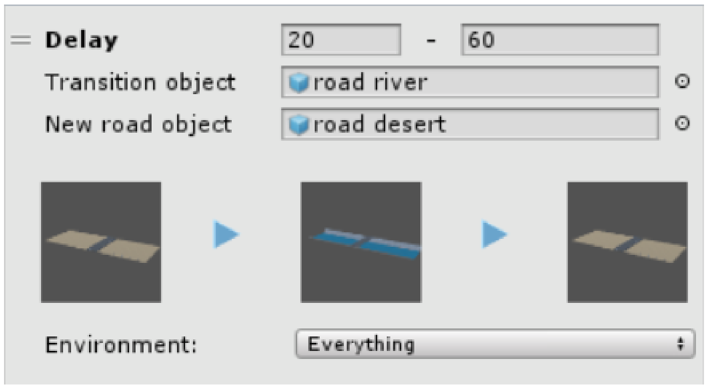
Manager script

The manager script is very important to the game, since it handles the UI. Things like count down, Game UI, game over menu and resetting the game using PlayerPrefs. The manager script is the only script which inspector is controlled by an editor script. I think using an editor script is handy to delete PlayerPrefs, because you don’t have to run the game that way. The manager holds the high speed variable too, but it doesn’t really use it, it just gives it to the scrolltexture script.

Road manager

Since version 2.0, each scene has its own road manager. This manager takes care of road transitions and roadside object transitions. To create a custom endless road, first create some new road objects. To do this, please go to prefabs -> environment -> road objects and drag one into your scene. You can now add your own textures to it. Please make sure your textures have the same tiling as the default textures for them to scroll correctly. Please add the new road object to your prefabs so you can use it.

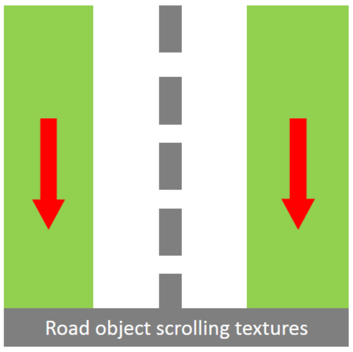
If you click the road manager, you should see the road transitions script. This is where you control your roads. Each block in the list represents a road transition:

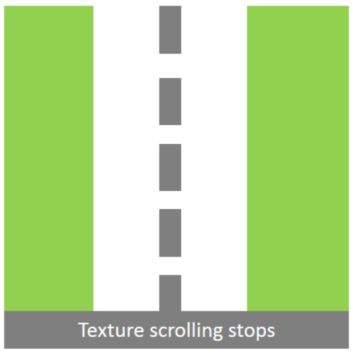


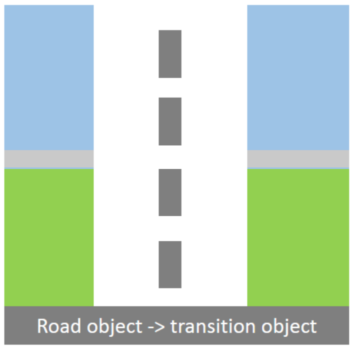
Delay 20 - 60
Transition object: road river
New road object: road desert
Environment: Everything

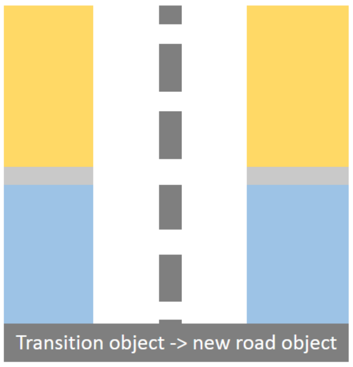
The delay has a min and max value so you can control its randomness. The delay in a road transition stands for the delay <u>before</u> the transition takes place. The transition object is the object in the middle of the road transition. For example, a river or a road crossing. The new road object is the object with the actual scrolling textures placed after the road transition. The environment filter uses data from your environment spawner objects so you can control your roadside objects after each transition. With the small handles, you can reorder your transitions.

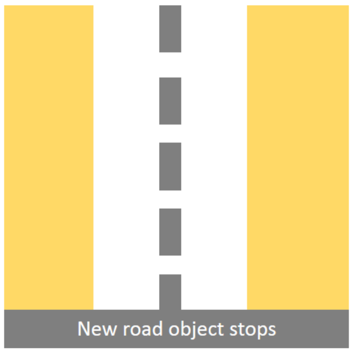
Now, for the road to work properly, it’s very important to understand the difference between endless road objects and single, non-scrolling road objects. Basically, what happens is this:

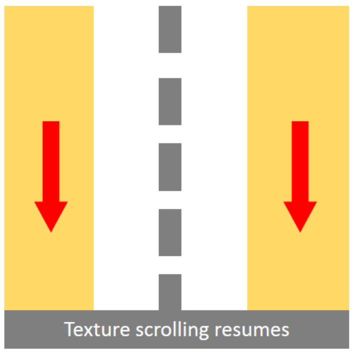






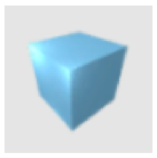






As you can see, only the grass object and the desert object have scrolling textures. The river object is moving anyway, so scrolling textures would look way to fast. To make your textures scroll (only textures on scrolling road objects), please see the script above the road manager, called ‘scroll texture’. By adding your materials (with textures) to the materials array, they will scroll properly. The water array is for river/sea water so it also moves to the right.

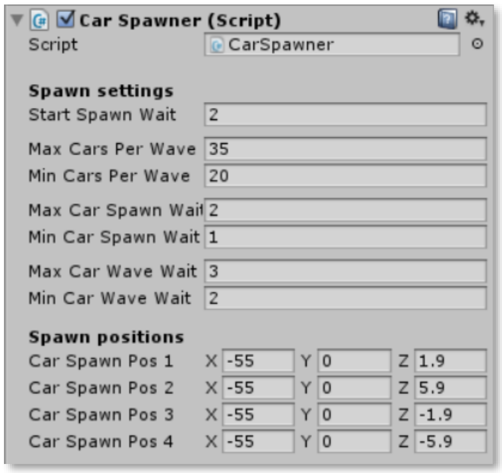
Please note: You might see these images in the road manager: It’s not a problem, it just means the asset previews didn’t load.



Spawning cars & environment

The car and environment spawner work almost the same. The biggest difference is the spawning position. For the environment, there are two main positions (each side of the road) with a random spawn range to create some variety. The car spawner has 4 exact positions (each track) without a random range. With the min and max variables in the inspector, you can fully control the wave spawning and in the environment spawner, there's a separate variable called bridge. That's because bridges are always spawned at the same position (in the middle).

Since version 2.0, you can also type in a boat name. This is the name of any boat object you add to the roadside objects. By using this name, boats will be rotated differently so they don't point at the road.



Car Spawner (Script)

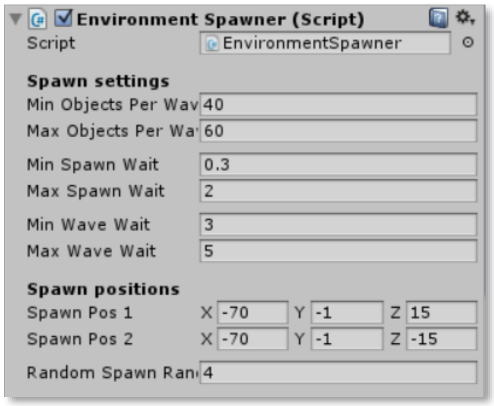
Script	CarSpawner

Spawn settings

Start Spawn Wait	2
Max Cars Per Wave	35
Min Cars Per Wave	20
Max Car Spawn Wait	2
Min Car Spawn Wait	1
Max Car Wave Wait	3
Min Car Wave Wait	2

Spawn positions

	X	Y	Z
Car Spawn Pos 1	-55	0	1.9
Car Spawn Pos 2	-55	0	5.9
Car Spawn Pos 3	-55	0	-1.9
Car Spawn Pos 4	-55	0	-5.9



Environment Spawner (Script)

Script	EnvironmentSpawner

Spawn settings

Min Objects Per Wav	40
Max Objects Per Wa	60
Min Spawn Wait	0.3
Max Spawn Wait	2
Min Wave Wait	3
Max Wave Wait	5

Spawn positions

	X	Y	Z
Spawn Pos 1	-70	-1	15
Spawn Pos 2	-70	-1	-15

Random Spawn Ran	4

Creating new race scenes

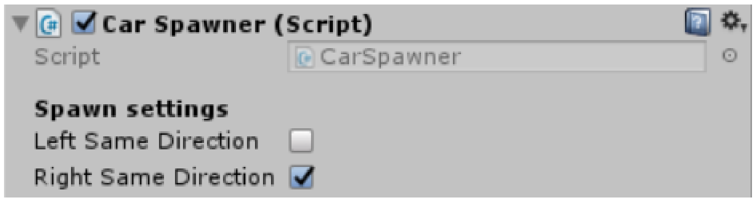
To make your own endless game scenes, just copy one of the included scenes. Then use the road manager to create your endless roads and add your own cars and environment to the game manager. If you want to use your scene in the game, you'll need to add it to the build settings and add an extra button to the locations panel in the garage. You will probably also need to add the default scenes to your build settings.

Crashing

The drivable cars (cars controlled by the player) have two ways of crashing and making the game over screen appear. First, they have a trigger on the front which immediately causes a crash. Secondly, they count the amount of collisions with other cars and after 5 collisions; they set the smoke effect active. If the smoke effect is active and the car collides again, it will crash.

Two directions

Since v1.4, cars can drive in two directions. Two enable/disable two directions simply use the checkboxes in the car spawner:


Car Spawner (Script)
Script CarSpawner

Spawn settings
Left Same Direction ☐
Right Same Direction ✓

Left same direction means that all cars on the left hand side move in the same direction as the player and right same direction means that all cars on the right move in the same direction. So not checking 'left same direction' means all cars on the left drive towards the player. Disabling both options would make the game quite hard, since all cars would drive in the opposite direction of the player.

Unity Ads

If you're creating a mobile game, Unity Ads is an easy way to have ads in your game. The kit comes with basic Unity Ads integration. When players crash and you've enabled Unity Ads, players can choose to watch an add to continue. The car will blink for a few seconds and it will continue driving. Please follow these steps to enable ads:

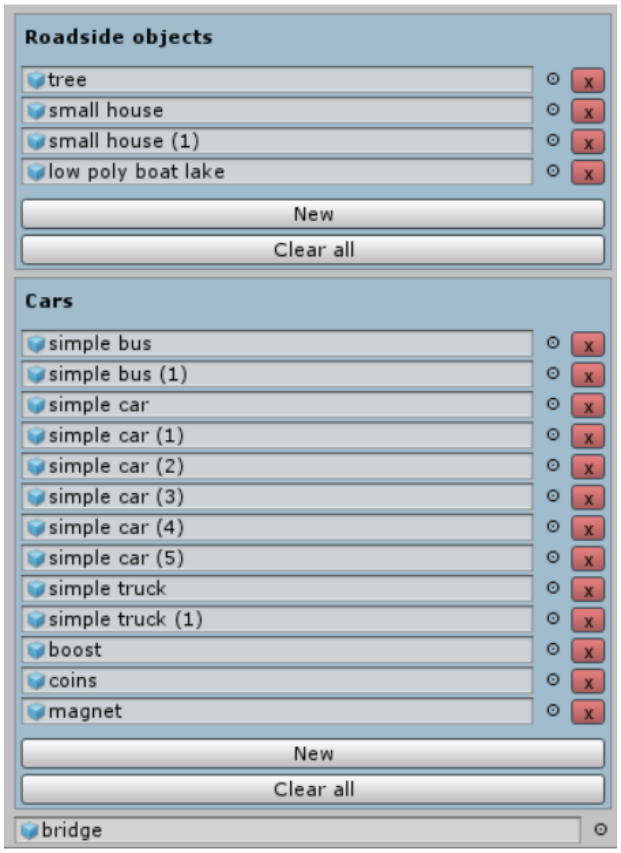
- First, make sure you switch your build platform to Android/IOS. This will automatically switch car controls as well.
- Now, go to window -> services (Ctrl + 0) and create a project ID.
- Enable ads in the ads window, select your platform and enable test mode.
- Under 'advanced' copy your GameID.
- In both racing scenes, go to the game manager and paste your GameID in the 'Unity Ads GameID' field.
- In the ads window, click 'go to dashboard', find your project, select the platform and press 'edit' to change ad settings (for the current system, it's best to use rewarded ads, since continuing after a crash is a reward).
- For each drivable car, choose if you want to show ads in car controls.

Please note:

1. **If you get a FileNotFoundException error, please go to the advanced ads settings, turn 'enable built-in ads extension' off and enable it again.**
2. **Don't forget to disable test mode before releasing your game.**

Object pooling

Since version 2.1, the kit contains object pooling. Basically, instead of removing and adding new cars and roadside objects all the time, spawned objects are stored in a list so they can be reused. Each prefab in the list has a list which stores the spawned objects of that type. For example, if the tree is spawned on the roadside it remembers where it came from and will be added to that list eventually. Then if the spawner decides to spawn another tree, it sees the tree in the list and it will use that one. If it wants to spawn another tree while all trees are in the game already, it will instantiate and use a new one.



Roadside objects

- tree
- small house
- small house (1)
- low poly boat lake

New
Clear all

Cars

- simple bus
- simple bus (1)
- simple car
- simple car (1)
- simple car (2)
- simple car (3)
- simple car (4)
- simple car (5)
- simple truck
- simple truck (1)
- boost
- coins
- magnet

New
Clear all

bridge

Object pooler

The object field below the roadside/car list contains your bridge prefab.

Conclusion

Thank you again for your support. In this document I tried to explain the Traffic Racing complete kit as clearly as possible. I hope you make great games with it. All code has been commented, so that should help for customization.

If you have any questions, you can always contact me via:

T3Dmake@gmail.com

If you like the kit, a review in the asset store would help a lot.