Guide DE  
dÉveloppement

Chronogame

20 janvier 2021

Cédric Lesueur

Chronostudio

Table des matières

[Introduction 2](#_Toc62999631)

[Ajout de véhicule 3](#_Toc62999632)

[Import du modèle 3](#_Toc62999633)

[Import des textures 3](#_Toc62999634)

[Création des matériaux 5](#_Toc62999635)

[Application des materials 7](#_Toc62999636)

[Nommage des parties 7](#_Toc62999637)

[Composants 9](#_Toc62999638)

[Parent racine (Nom\_du\_véhicule) 9](#_Toc62999639)

[Corps du véhicule (Body) 10](#_Toc62999640)

[Roues (Front/Back wheels) 11](#_Toc62999641)

[Caméra 13](#_Toc62999642)

[Ajout du véhicule à la liste 14](#_Toc62999643)

[Prefabs 14](#_Toc62999644)

[Liste des véhicules 15](#_Toc62999645)

# Introduction

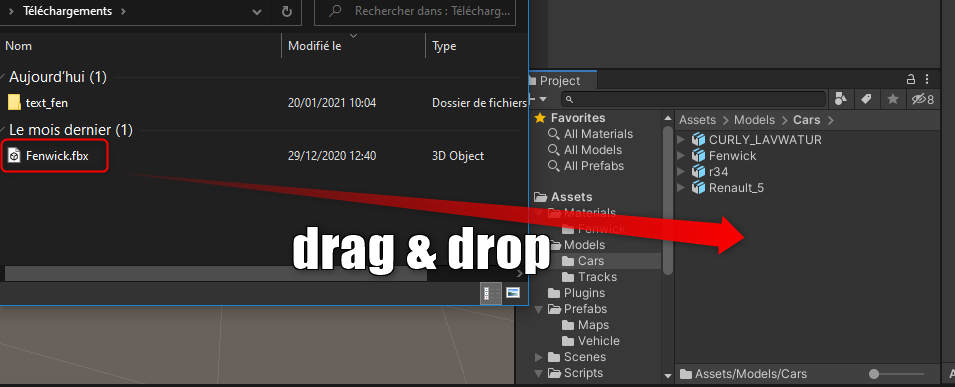
Ce document contient différents guides concernant le jeu Chronogame. Ils sont utiles pour le bon développement de celui-ci.

Les principaux guides décrivent l’ajout de modèle et comment les intégrer aux scènes.

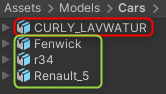
# Ajout de véhicule

## Import du modèle

Le modèle doit être au format FBX pour pouvoir être importé sur Unity. Une fois le modèle télécharger dans le bon format, il faut l’importer dans Unity. Un simple *drag & drop* de la fenêtre d’explorateur de fichier vers l’éditeur Unity suffit.



Pour des questions d’ordre et de propreté, le modèle est à importer dans le dossier *Assets/Models/Cars/*. Pour ces mêmes raisons de propreté, il faut lui donner un nom représentatif (ex : Fenwick.fbx).



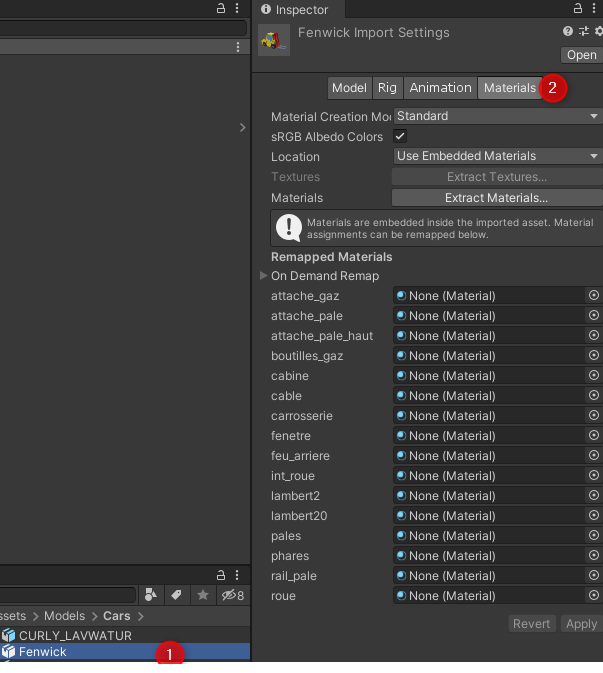
En rouge, la mauvaise manière de nommer un modèle.

En vert, la bonne manière de nommer un modèle.

## Import des textures

De la même manière que le modèle, les textures sont à importer dans Unity avec un format spécifique. Il faut demander aux graphistes les *base colors* correspondantes à la map, celles-ci doivent être au format PNG.

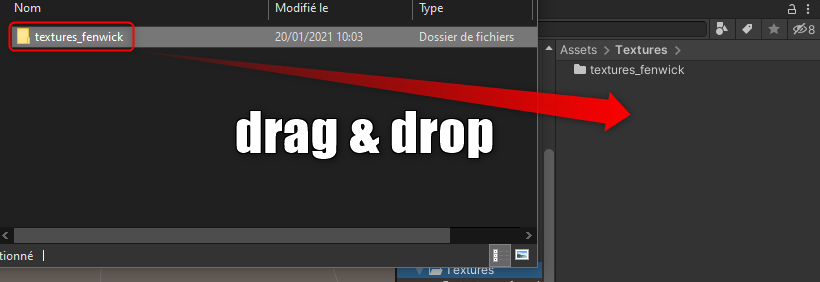
Pour savoir à quoi ressemble la map des textures, il suffit de sélectionner (clic simple)[1] le véhicule dans le dossier où il a été importé et d’aller dans l’onglet *Materials*[2]. Une map ressemble à quelque chose du genre :



Si comme ici, il y a quelque partie avec un nom qui n’est pas facilement identifiable (ex : lambert2), il faut découvrir à quelle partie du véhicule cela correspond. Le plus simple est de demander directement à l’auteur du modèle. Si aucun des noms n’est explicite, demandez à l’auteur de modifier ça.

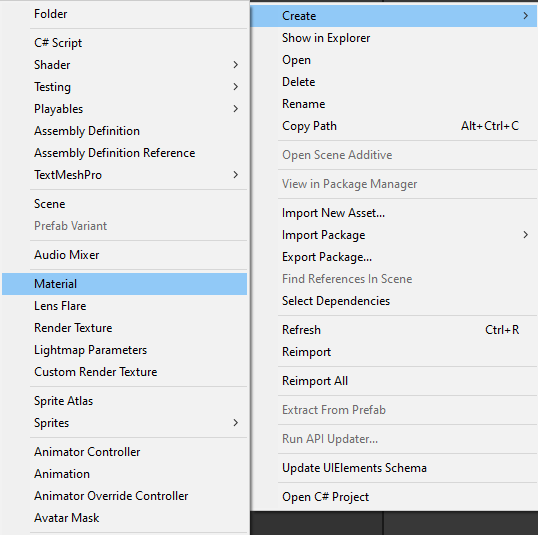
Pour importer les textures (base colors), il suffit de faire un *drag & drop* du dossier de textures comme pour le modèle. La seule différence est qu’il faut le faire dans le dossier *Assets/Textures/* comme montré ci-dessous. Au niveau du nommage de ce dossier, il faut suivre l’exemple suivant :

Pour le fenwick, le dossier de textures s’appelle *textures\_fenwick*

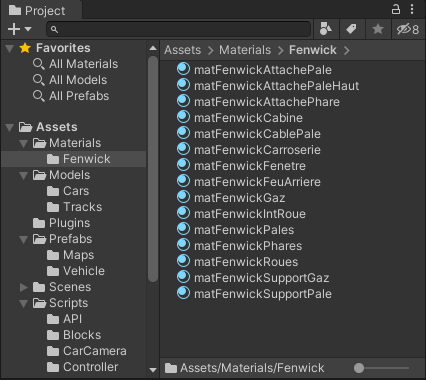


## Création des matériaux

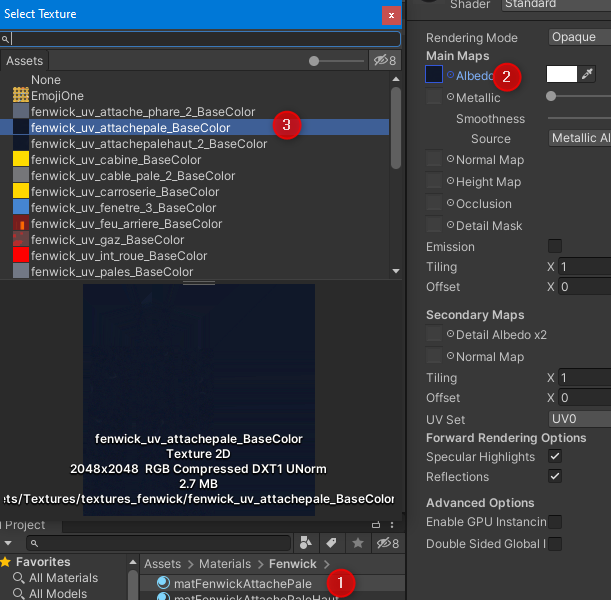
Maintenant que les textures sont importées, il faut créer les matériaux correspondants. Pour cela, on se place dans le dossier *Assets/Materials/{nom\_du\_véhicule}*. A partir de la il faut faire un *clic droit > Create > Material*



Une fois le material créé, on lui donne un nom logique, par exemple pour le material des phares du fenwick, *matFenwickPhare*. Une fois tous les materials créés, on devrait obtenir un résultat similaire à celui-ci



Il faut maintenant appliquer nos textures aux materials. Rien de plus facile. Il suffit de sélectionner le material, de cliquer sur le bouton de choix *albedo* (le truc en forme de téton) et de sélectionner la texture voulue.



On peut maintenant appliquer les materials au modèle !

## Application des materials

Pour appliquer les materials au modèle, on retourne au niveau de la map.

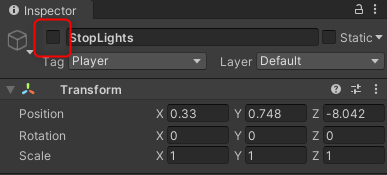
## Nommage des parties

La première partie concernant le nommage des parts se fait juste sur les 5 parties principales : le corps, la roue antérieure gauche, la roue antérieure droite, la roue postérieure gauche et la roue postérieur droite.

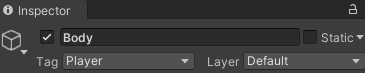
Le résultat devrait ressembler à cela pour l’instant :



Il faut bien que les lumières (Lights) soient désactivées :



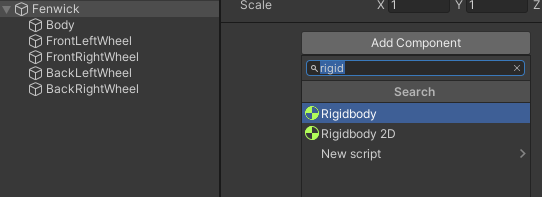
Et il ne faut pas oublier de mettre le tag du *Body* à *Player*.



## Composants

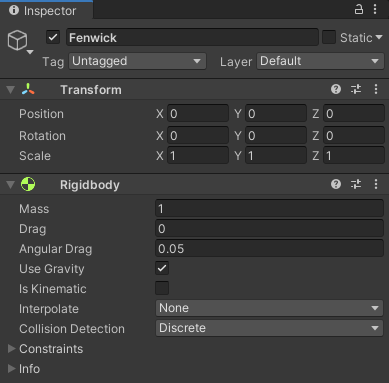
### Parent racine (Nom\_du\_véhicule)

#### Rigidbody

Le premier composant est à mettre sur le parent du véhicule. Il faut lui mettre un rigidbody

*Clic gauche sur le parent (dans ce cas fenwick) > Add component dans l’éditeur > Rigidbody*

Le composant est maintenant rattaché à l’objet



Il ne reste plus qu’à changer la masse (Mass) du véhicule. Dans notre cas on la met à 2000.

N.B : Sur Unity une unité de masse équivaut à 1kg.

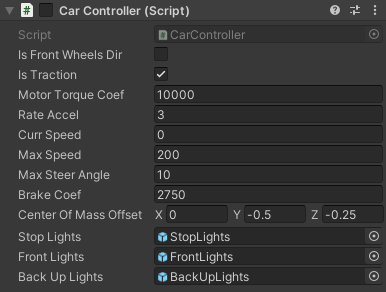
#### Scripts

Au niveau du script il faut lui rattacher le *CarController.cs* qui se trouve dans le dossier *Scripts/Controller/*.

Il faut maintenant paramétrer les données du script en fonction du résultat souhaité :



Dans notre cas, le résultat est le suivant :



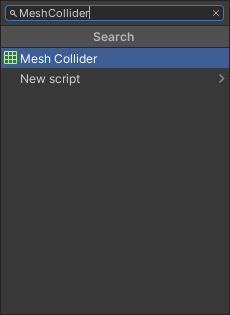
Il ne faut pas hésiter à tester et remodifier les valeurs pour obtenir le résultat escompté. Dans ce cas aussi il faut désactiver le script.

### Corps du véhicule (Body)

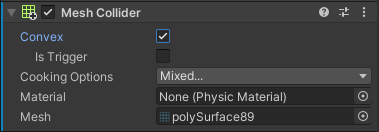
#### Collider

Il faut sélectionner l’objet que l’on a renommé *Body*. Et on y ajoute un composant *MeshCollider*

*Clic gauche sur l’objet > Add component > MeshCollider*



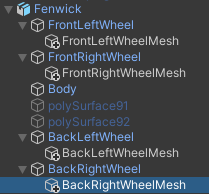
Dans le mesh collider, il faut cocher la case *Convex*.



### Roues (Front/Back wheels)

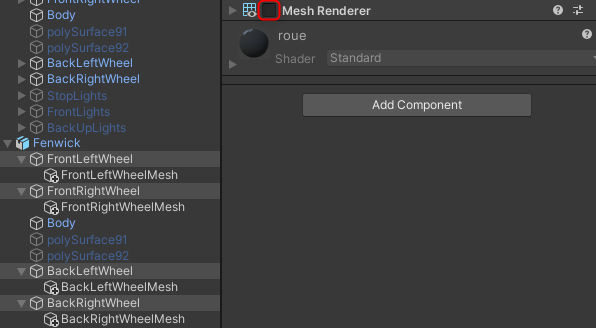
#### WheelCollider

Au niveau des roues, c’est un peu plus complexe. Il faut copier les roues et les coller sur leur objet roue.

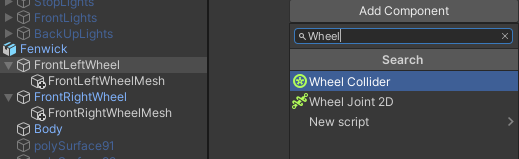


Et il faut bien renommer l’objet que l’on copie en ajoutant *Mesh* à la fin.

A partir de là, on désactive le *MeshRenderer* des objets **roue** qui ne contiennent pas *Mesh* dans leur nom.

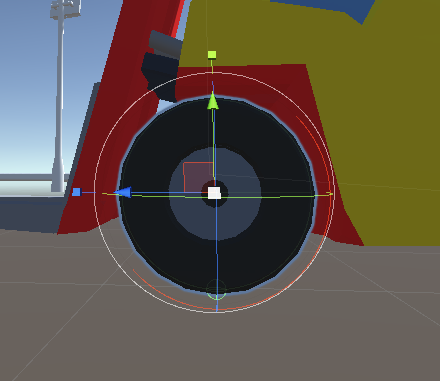


Les roues devraient encore être visibles. Sur ces mêmes objets, on ajoute (roue par roue) un *WheelCollider*.



*Clic gauche sur l’objet > Add component > WheelCollider*

A partir de là il faut paramétrer les colliders. Il faut aligner le collider avec les roues



Pour cela, on met la *Suspension Distance* à 0



Et on change le *Radius* pour obtenir un bon résultat

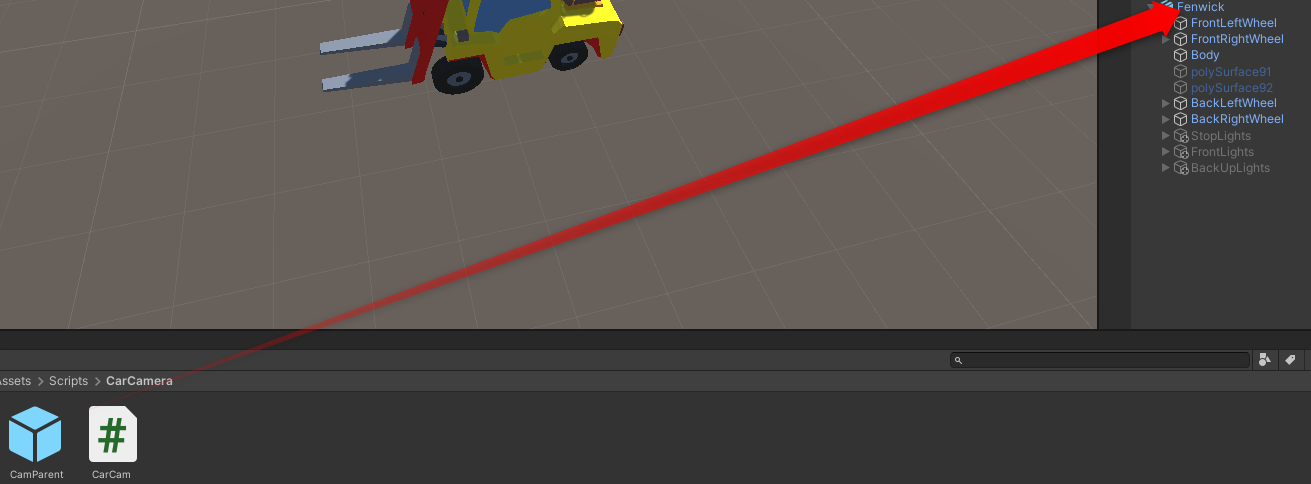


Pour les roues arrière il ne faut pas oublier de mettre la *Stiffness* à 2, sinon le véhicule va partir du cul

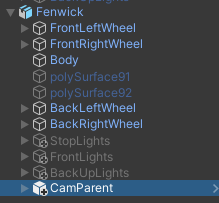


## Caméra

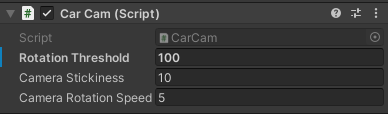
Pour la caméra il suffit de faire glisser l’objet *CamParent* sur le véhicule dans la hiérarchie. L’objet en question se trouve dans le dossier *Scripts/CarCamera/*.



Il devrait apparaitre dans la hiérarchie



Il faut maintenant paramétrer la caméra avec ces valeurs (valeurs susceptibles de changer par la suite)



## Ajout du véhicule à la liste

Dernière étape, il faut ajouter le véhicule maintenant créer au dossier des *Prefabs* pour commencer pour à la liste des véhicules.

### Prefabs

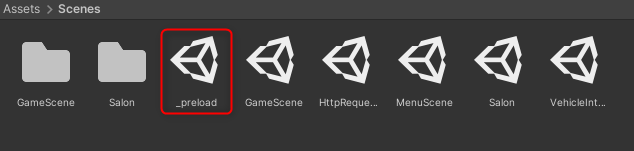
Pour l’ajouter aux *Prefabs* il suffit de glisser et déposer le véhicule de la hiérarchie au dossier *Prefabs/Cars/*.



Et lorsque l’on nous le demande, cliquer sur *Original Prefab*. On peut maintenant supprimer le véhicule de la scène.

### Liste des véhicules

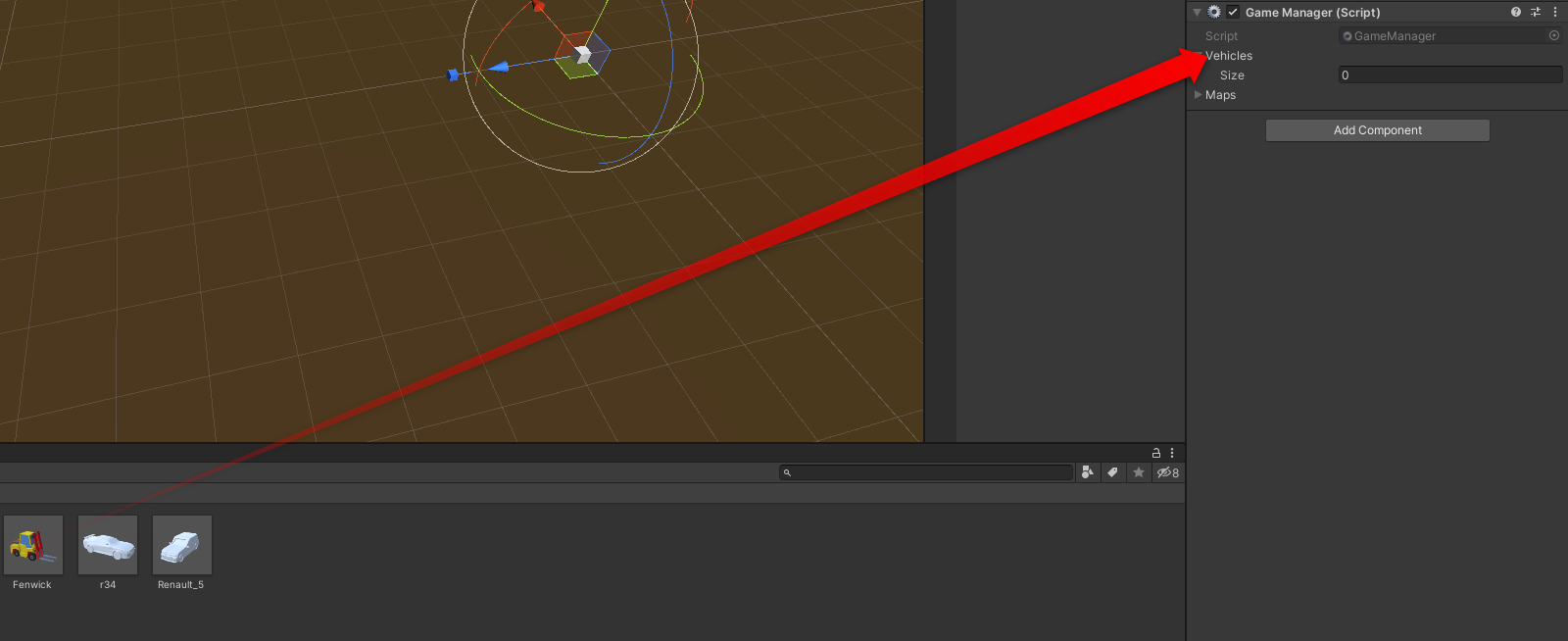
Et passer à la scène *\_preload* dans le dossier *Scenes/*.



Sélectionner l’objet *\_app* dans la hiérarchie.



Et glisser le *prefab* du véhicule **sur** la liste véhicule (où il y a écrit véhicule dans l’éditeur).



Le véhicule sera maintenant pris en compte au chargement du jeu.

