Tout ce qui est écrit en bleu correspond au fichier route.

C'est un fichier .csv avec le nom que l'on veut par exemple : route01.csv

La première chose à savoir c'est que le Rail 0 est le rail ou circule le train et qu'il existe par défaut dans le fichier route, on n'a pas besoin de le définir, il commence au point 0,0,0 de la route, cela correspond a l'écriture X,Y,Z dans les fichiers de développement (route-csv.pdf).
X est l'axe horizontal perpendiculaire à la voie, le négatif (-1) c'est vers la gauche et le positif (1) c'est vers la droite, le point 0 correspond au centre de la voie.
Y est l'axe vertical à la voie, le négatif (-1) c'est vers le bas et le positif (1) c'est vers le haut, le point Y=0 correspond au dessus du rail.
Z est l'axe de la voie, le négatif (-1) c'est vers l'arrière et le positif (1) c'est vers l'avant, le rail démarre au point 0 le suivant au point 25 et ainsi de suite tous les 25 m.
Pour le voir il faut avoir créé l'image du rail dans un fichier objet rail0.csv et le définir avec la commande With Structure dans le fichier Route par exemple :

With Structure
.Rail(0).Load (Objets\track\Track0.csv)
.Rail(1).Load (Objets\track\Heurtoir_0d.csv)
.Rail(2).Load (Objets\track\Curve400GW.csv)
La commande .Rail(n) charge les rails que l'on va utiliser.

C'est un exemple, j'ai mis mes fichiers dans le répertoire *Objets*, on peut le nommer autrement souvent en rapport avec le nom de la route, dans les objets des routes Anglaise les rails sont dans un répertoire *track* que j'ai gardé pour mes rails mais on peut aussi le nommer autrement, (*chacun fait comme il veut pour les noms des répertoires*).
La commande With Structure charge les objets pour la route, mais le Rail 0 apparaît après la commande With Track c'est à partir de là que la route commence vraiment.

With Track
0, (*ici commence la voie au Km 0*)
.Sta Départ; 07.5900; 08.0000; ; -1; 1; ; (*commande de la gare, voir fichier routecsv.pdf*)
.RailType 0;1 (*C'est le rail 0 avec l'objet Heurtoir_0d.csv de la commande .Rail(1) de With Structure*)
C'est pour une voie simple, si on a une voie double on va rajouter le rail 1.
.RailStart 1;4.34;0;1 (*C'est le rail 1 a 4.34 m à droite du rail 0 et au même niveau 0, voir fichier routecsv.pdf*)
Pour d'autre voie on rajoutera d'autre rails à gauche ou à droite du rail 0, tous les rails en plus sont défini par rapport au rail 0, et toujours les définir à partir de ce rail car il existe toujours.

25, (*on est à 25 mètres*)
.RailType 0;0 (*On change de type de rail le rail0 avec l'objet Track0.csv de la commande .Rail(0) de With Structure*)
.RailType 1;0 (*On change le type du rail 1*)
C'est le même rail à 50, 75, ainsi de suite tous les 25 m, jusqu'à ce que l'on change de type de rail.
Pour créer une courbe on écrit par exemple à 100 m :

100, (*on est à 100 mètres*)
.Curve -400; 50 (*Courbe vers la gauche d'un rayon de 400 m et d'un devers de 50*)

.RailType 0;2 (*changement de type de rail le rail0 par un rail courbe*)
.RailType 1;2 (*changement de type de rail le rail0 par un rail courbe*)

Pour une courbe à droite c'est .Curve 400;50 et pour la fin de la courbe c'est .Curve 0, et on repart sur un rail droit, pour les aiguillage quand le train file droit c'est comme un rail droit (Curve 0;0) quand le train tourne c'est un courbe (.Curve 145;0) et on défini le type de rail en fonction du choix.
Pour les gares il y a la commande .Sta qui donne le nom de la gare, l'heure d'arrivée et de départ, l'indication de quel coté les portes s'ouvrent, la durée d'arrêt etc ...

Maintenant que l'on a le rail on va s'occuper du sol avec la commande .Ground
On charge les fichiers objets ground que l'on a créé dans With structure à la suite des rails.

With Structure
.Ground(0).Load (Objets\ground\ground0.csv)
.Ground(1).Load (Objets\ground\ground1.csv)
La commande .Ground(n) charge le sol que l'on va utiliser.
On la place en début, exemple :
With Track
0, (*ici commence la voie au km 0*)
.Ground 0
La commande va se répéter tout les 25 mètres, (c'est la distance par défaut que l'on a choisi), jusqu'à ce que l'on change de .Ground(n), et le sol est défini à -0.3 voir image.
100, (*on est à 100 mètres*)
.Ground 1
Et ainsi de suite.

Maintenant que l'on a le rail et le sol on va s'occuper des quais de la gare avec la commande .Form
On charge les fichiers objets pole que l'on a créé dans With structure à la suite des rails et des sols.

With Structure
.FormL(0).Load (Objets\station\FormL-0.csv)
.FormCL(0).Load (Objets\station\FormCL-0.csv)
.FormR(0).Load (Objets\station\FormR-0.csv)
.FormCR(0).Load (Objets\station\FormCR-0.csv)
Les commandes .FormL(n) .FormCL(n) .FormR(n) .FormCR(n) charge les quais de gare.
On la place au niveau des gares, exemple :

With Track
0, (*ici commence la voie au km 0*)
.Form 0;L;0;0; (*quai positionné sur le rail0 et à gauche*)
.Form 1;R;0;0; (*quai positionné sur le rail1 et à droite*)
J'ai un quai sur la voie 0 à gauche et un quai sur la voie 1 à droite.

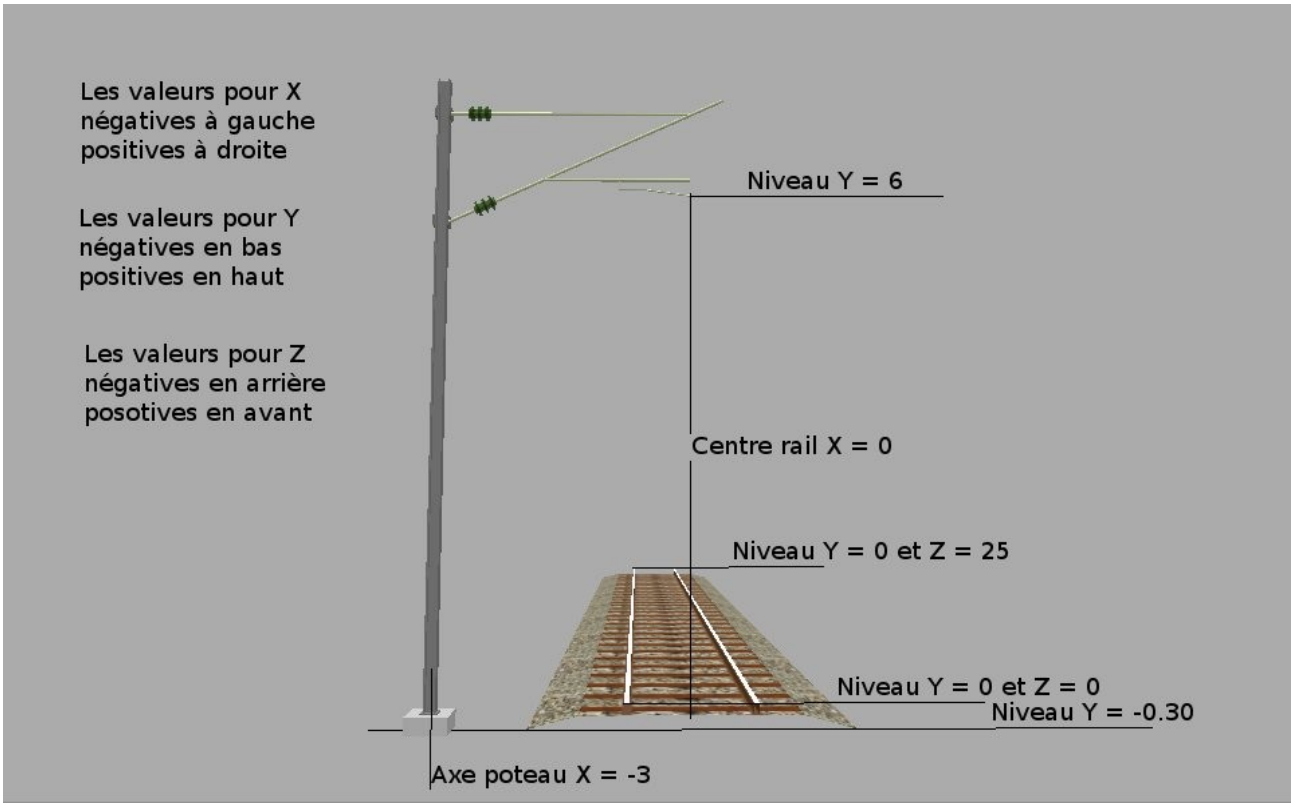
Maintenant que l'on a le rail, le sol et les quais on va s'occuper des poteaux pour les caténaires avec la commande .Pole, le fichier pole que j'ai créé a pour point de construction l'axe de la voie.
On charge les fichiers objets pole que l'on a créé dans With structure à la suite des rails et des sols.

With Structure
.Pole(0; 0).Load (Objets\poles\pole-0g-3d.csv)
(La commande .Pole(n;n) charge les poteaux des catenaires.)

On la place en début, exemple :

```
With Track
0,                          (ici commence la voie au km 0)
.Pole 0;0;0;50;0            (Poteau simple sur le rail0 et à gauche)
.Pole 1;0;1;50;0            (Poteau simple sur le rail1 et à droite)
```

Dans l'exemple ci dessus les poteaux sont placé tous les 50 m, soit tous les deux rails. Si au lieu de 50 il y avait 25 c'est tous les 25 m.



Maintenant on va s'occuper de charger les murs .Wall que l'on va utiliser, on les fait d'une longueur de 25 pour tout un rail ou plus court s'il s'arrête avant la fin d'un rail.

```
With Structure
.WallL(0).Load (Objets\wall\wallL0.csv)      (Mur du coté gauche d'un rail)
.WallR(0).Load (Objets\wall\wallL0.csv)      (Mur du coté droit d'un rail)
.WallL(1).Load (Objets\wall\wallL0.csv)      (Autre mur du coté gauche d'un rail)
.WallR(1).Load (Objets\wall\wallL0.csv)      (Autre mur du coté droit d'un rail)
```

On le place en début d'un rail qui longe un mur, exemple :

```
With Track
200,                        (ici on va commencer le mur a 200m)
.Wall 0;-1;0                (Mur WallL(0) placer à gauche du rail0)
.Wall 1; 1;0                (Mur WallR(0) placer à droite du rail1)
```

Le mur placé se répète pour chaque rail qui suit celui ou il est indiqué jusqu'à une commande .Wall 0;-1;1 qui change de mur et il ne prend fin qu'avec la commande .WallEnd

```
300,                        (ici on va arrêter le mur a 300m)
.WallEnd 0                  (ici on arrête le mur du rail0)
.WallEnd 1                  (ici on arrête le mur du rail1)
```

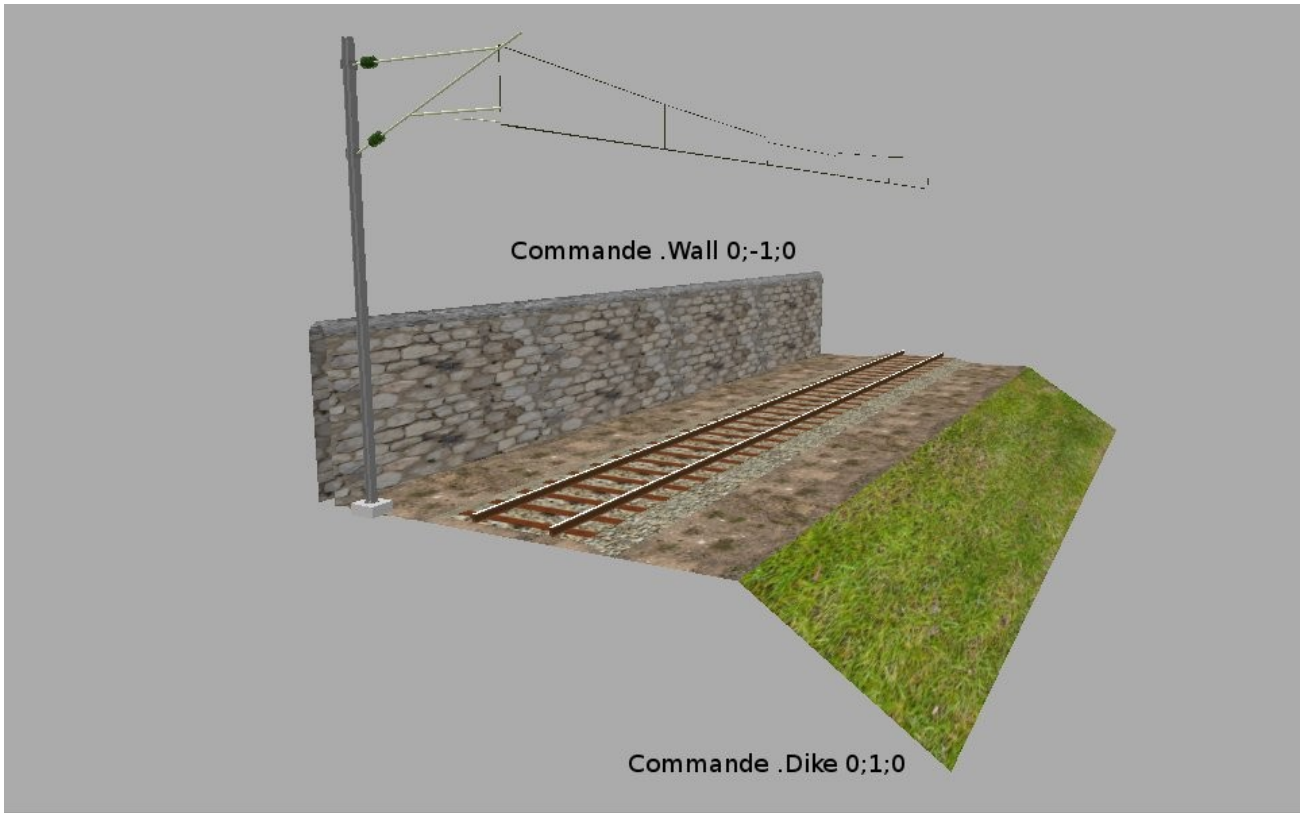
Ensuite il y a les Talus on va les charger, ils ont les même propriété que les murs.

```
With Structure
.DikeL(0).Load (Objets\wall\wallL0.csv)      (Talus du coté gauche d'un rail)
.DikeR(0).Load (Objets\wall\wallL0.csv)      (Talus Mur du coté droit d'un rail)
.DikeL(1).Load (Objets\wall\wallL0.csv)      (Talus Autre mur du coté gauche d'un rail)
.DikeR(1).Load (Objets\wall\wallL0.csv)      (Talus Autre mur du coté droit d'un rail)
```

On le place en début d'un rail qui longe un talus, exemple :

```
With Track
200,                                 (ici on va commencer le mur a 200m)
.Dike 0;-1;0                         (Talus DikeL(0) placer à gauche du rail0)
.Dike 1; 1;0                         (Talus DikeR(0) placer à droite du rail1)
```

Comme pour les murs ils se répètent tous les rails suivant pour mettre fin au talus il faut la commande .DikeEnd comme pour changer de talus il faut une nouvelle commande .Dike



Maintenant il faut ajouter des objets à la suite dans With Structure :

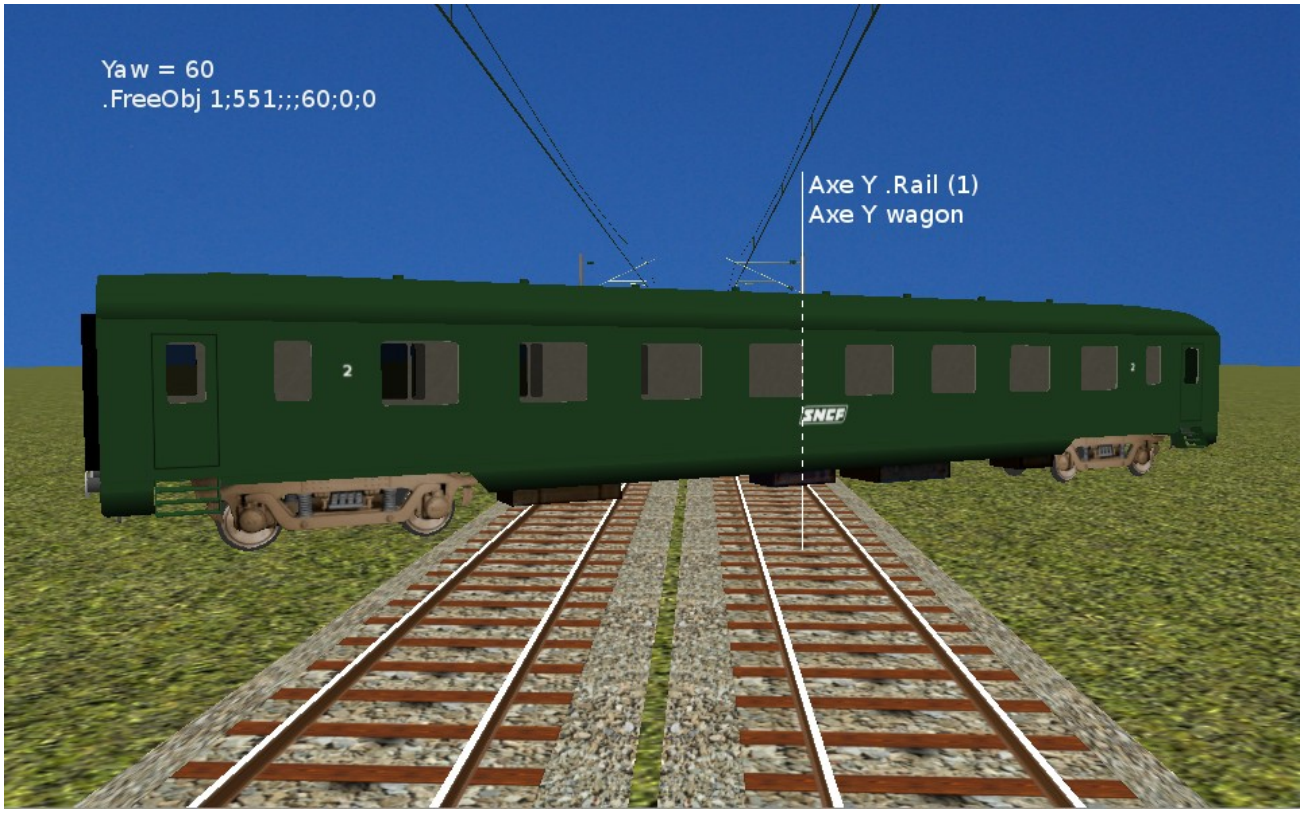
```
With Structure
.FreeObj(0).Load (Objets\poles\catenaire-50.csv)
.FreeObj(1).Load (Objets\poles\catenaire-25.csv)
.FreeObj(2).Load (Objets\signals\Halte-4.csv)
.FreeObj(3).Load (Objets\signals\Halte-6.csv)
.FreeObj(4).Load (Objets\signals\Halte-8.csv)
.FreeObj(5).Load (Objets\signals\Halte.csv)
```

Plus on veut améliorer le paysage de notre route plus on aura d'objets à charger et placer.
Plus le fichier Route sera lourd pour la simulation.

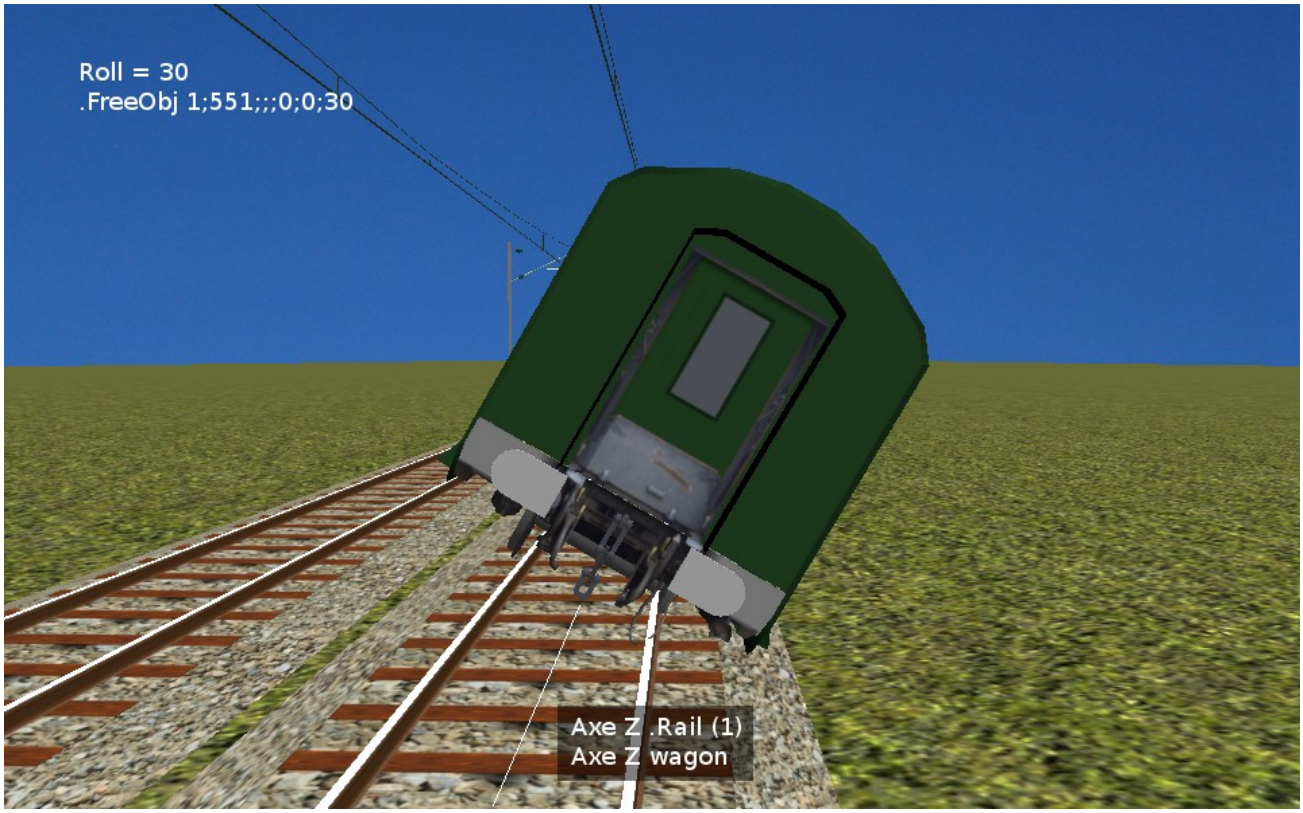
On va placer les fichiers objets sur la voie après la commande With Track

With Track
0, (*ici commence la voie au km 0 valeur de Z*)
.FreeObj 0;0;;;0; (*ici place un objet par rapport au rail0 au km 0*)
.FreeObj 1;0;;;0; (*ici place un objet par rapport au rail1 au km 0*)
On peut placer les objets par rapport à n'importe quel autre rail que le rail0, mais tout est finalement placer par rapport au rail0 puisque tous les autre rails sont placés par rapport au rail0, l'objet ne se répète pas.
Dans l'exemple ci dessus on place l'objet au centre du rail, pour placer un objet a gauche c'est la commande .FreeObj 0;39; -42;-0.2;0; c'est donc X = -42 et Y = -0.2 cette commande s'écrit :
.FreeObj *RailIndex; FreeObjStructureIndex; X; Y; Yaw; Pitch; Roll*
RailIndex c'est l'index du rail, 0 pour le .Rail(0) 1 pour le .Rail(1) et ainsi de suite.
FreeObjStructureIndex c'est l'index de l'objet, 0 pour le .FreeObj(0) 1 pour le .FreeObj(1) et ainsi de suite.
X C'est la distance de l'objet par rapport au rail valeur positive coté droit et valeur négative coté gauche,.
Y C'est la distance de l'objet par rapport au rail valeur positive au dessus et valeur négative au dessous du rail.
Placé à la distance Z = 0, 25, 50, 75, etc l'objet est placé au début d'un rail sinon il faut rajouté une valeur différente comme :
10,
.FreeObj 0;0;;;0; (*ici place un objet par rapport au rail0 a Z = 10m*)
.FreeObj 1;0;;;0; (*ici place un objet par rapport au rail1 a Z =10m*)
Yaw C'est l'angle donné à l'objet par rapport à l'axe Y du rail et au point 0,0,0 de l'objet.
Pitch C'est l'angle donné à l'objet par rapport à l'axe X du rail et au point 0,0,0 de l'objet.
Roll C'est l'angle donné à l'objet par rapport à l'axe Z du rail et au point 0,0,0 de l'objet.

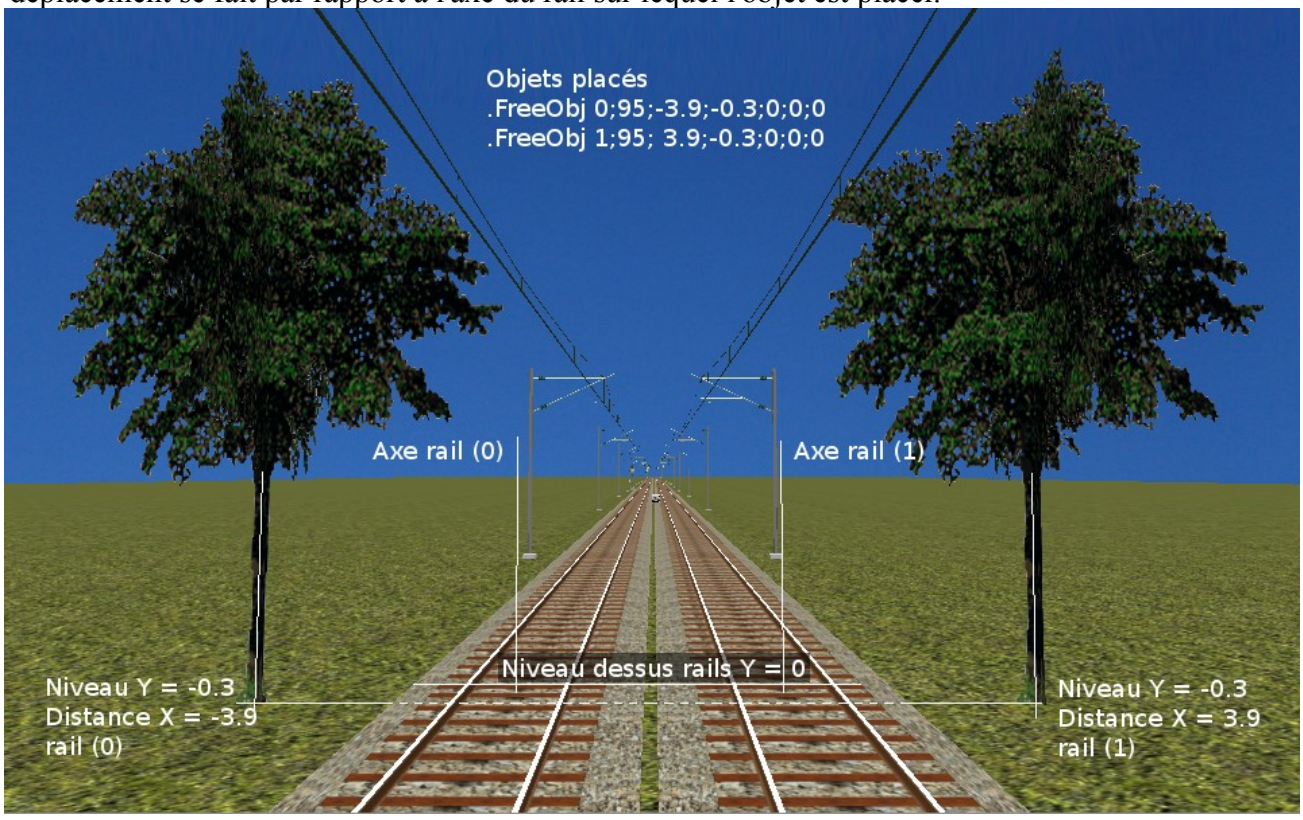
Voir les images ci-dessous.


Yaw = 60
.FreeObj 1;551;;;60;0;0
Axe Y .Rail (1)
Axe Y wagon
SNCF


Pitch = 30
.FreeObj 1;551;;;0;30;0
SNCF
Axe X .Rail (1)
Axe X wagon



Des images parlent plus qu'un long discours, l'objet est placé à X = 0 et Y = 0, ne pas oublier que le déplacement se fait par rapport à l'axe du rail sur lequel l'objet est placer.



Avec cela vous pouvez faire une ligne de chemin de fer simple, c'est un début car il faut tester ce que l'on fait avec le logiciel RouteViewer pour voir si cela correspond avec ce que l'on veut comme

résultat.

Maintenat pour aller plus loin il faut lire le fichier route-csv.pdf.