Train Controller (Railroad System)
User Manual v3.3

WSM GAME STUDIO

Doing all the hard work for you!

2019

SUMMARY

1. Intro

Thank you for purchasing “Train Controller (Railroad System)”!

This package contains all that you need to build a simple and functional railroad (models, scripts and SFX).

More models may be included in the future and/or sold separately as extensions (Addons Available).

It’s really simple to use and customize.

Follow this tutorial or watch it on youtube if you like: Youtube Tutorial

2. Update Guidelines v3.3

Version 3.3 has some structural changes on locomotive and wagons physics. These changes were made to increase train movement stability.

If you haven't installed any previous versions of this package on your project, or have completely removed any previous versions, then you don't need to worry about this section.

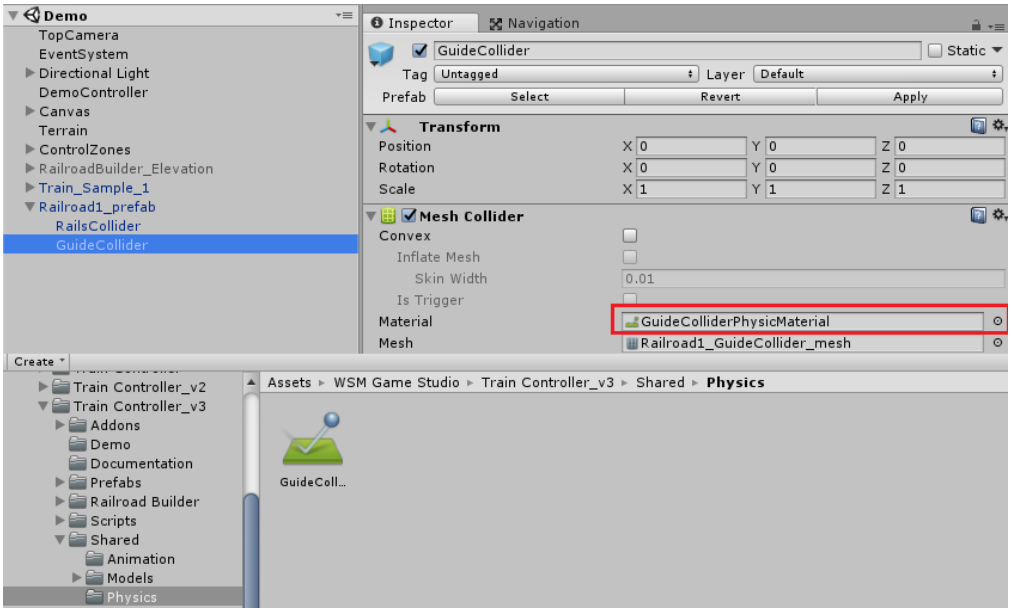
But, if you have already installed any previous versions on your project and downloaded version 3.3 as an **update** inside the Unity Editor. Then, you may need to check your **custom prefabs**, since the Unity Editor will **not** update custom prefabs automatically.

For custom Train prefabs created on previous versions, it's recommended to replace the locomotive and all wagons to make sure the new changes will apply correctly.

If you have downloaded any Additional Wagons, don't forget to download the corresponding updated version of those wagons before replacing them on your custom train prefabs.

You can also use the new Custom Wagons Creator to rebuild your custom locomotive and wagons prefabs. This will ensure that the custom wagons and locomotives are setup correctly.

For custom **baked** railroads and/or rail segments created on previous versions, you need to apply the "GuideColliderPhysicMaterial" to your custom rails "GuideCollider" child object.

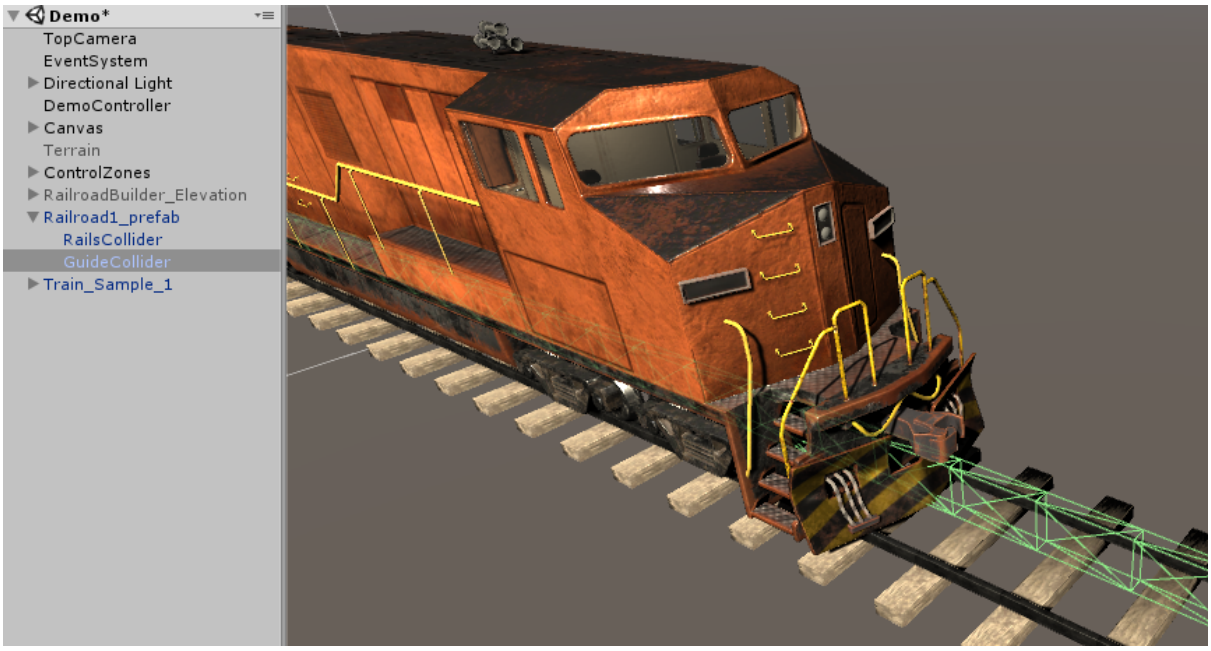


3. How to Use This Asset

In this section you will learn how this asset works and how to use it based on the sample prefabs.

3.1. How it Works

This asset is focused on **physics based train movement**. It works by using guide colliders to keep the train on rails, while applying forces to simulate acceleration, brakes and wagons connections.



The rails are composed by colliders tagged as “Rails” and guide colliders that composes the path that the train will follow.

The TrainController script is attached to the locomotive controls acceleration, brakes, SFX, doors, lights and wagons.

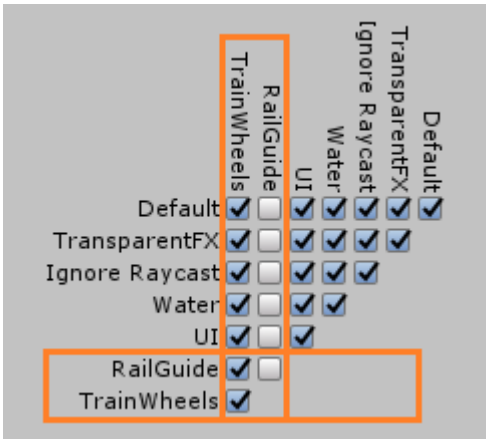
The wagons are connected by Joints and inherits locomotive movement and commands while connected.

3.2. Recommended Layers (Optional)

The package is ready to use as it is. Although, to avoid unwanted collisions with the rails guide colliders, it is recommended to customize the Unity Collision Matrix.

Go to “**Edit > Project Settings > Tags and Layers**”, add new layers for the rail guides and train wheels (Ex: RailGuide and TrainWheels). Change the layers of the train wheels and rail guides of the prefabs.

Go to “Edit > Project Settings > Physics” and change the collision matrix so the “RailGuide” layer will collide **only** with the ‘TrainWheels” layer.

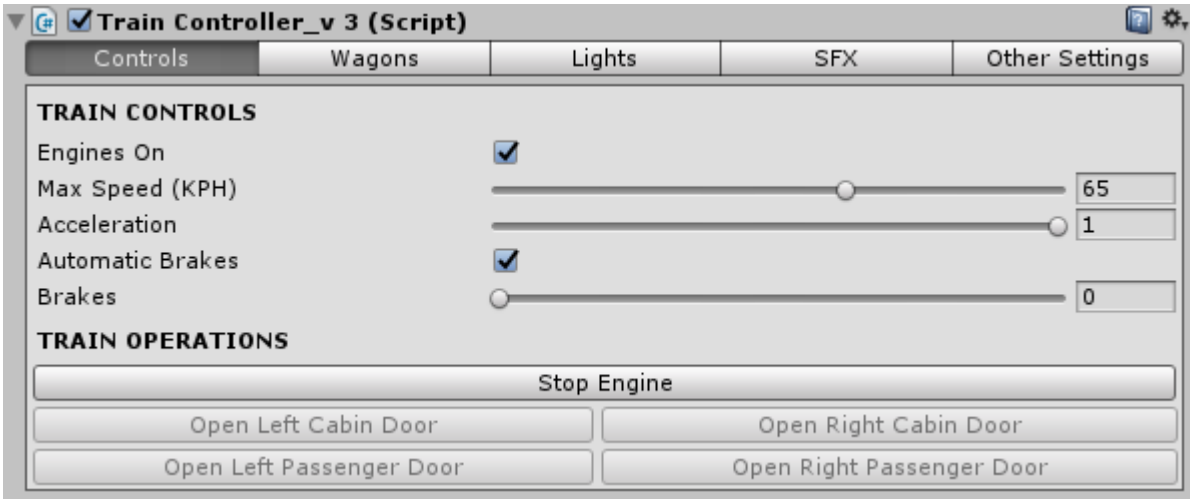


For more info about the collision matrix, check the [layer-based collision] official documentation.

3.3. Sample Prefabs (Quick Start)

This package includes a ready to use railroads and train sample prefabs. These prefabs can be seen in action at the demo scenes included in the project.

If you wish to use these prefabs on your game, all you need to do is drag and drop a railroad and a train to your scene. You can also customize speed and acceleration in the TrainController script attached to the locomotive.



Note: By default, the sample prefabs are configured as automated trains, if you wish to manually control your train, take a look at the player input section.

3.4. Locomotive & Wagons Prefabs

A train is composed by a locomotive and wagons. There are six locomotives and a set of wagons prefabs at the “Prefabs” folder.

More locomotives and wagons may be included in this package in the future and/or sold separately as extensions (Addons Available Here)

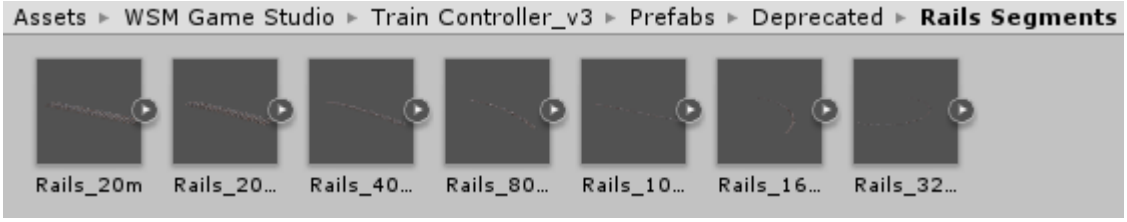
See the Building a Train section for more info on how to use these prefabs.

3.5. Legacy Prefabs (Deprecated)

Up to version 2.0, the railroads were composed by joined rail segments prefabs.

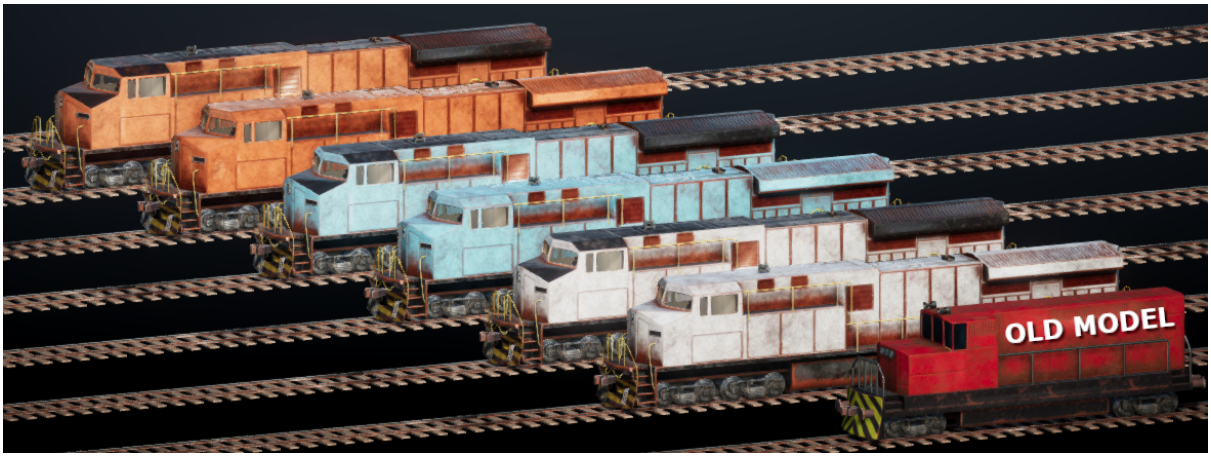
Starting with version 3.0, the railroads are now built using the **Railroad Builder**, which is much easier to use and give much more control over the railroad.

Although, the rail segments prefabs were kept at the “Deprecated > Rails Segments” folder for those whom bought any previous version of the package and wish to keep using them for any reason.

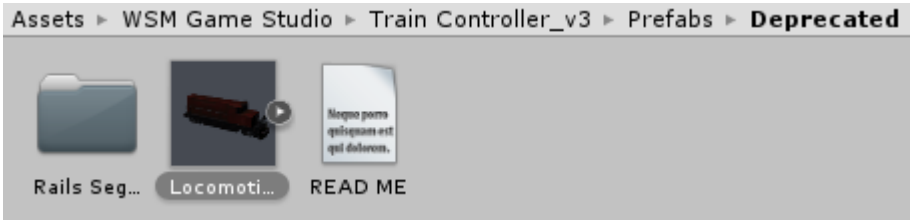


The **RailroadBuilder** prefab is now the official method for building railroads. See the Building a Railroad section for more info on how to use the Railroad Builder.

Starting with version 3.3, the default locomotive model was replaced by a more realistic model.



The old locomotive prefab can still be found inside the “Deprecated” folder.



Note: You can still use deprecated prefabs if you wish so, however, keep in mind that deprecated content will not be updated or guaranteed to be compatible with future versions.

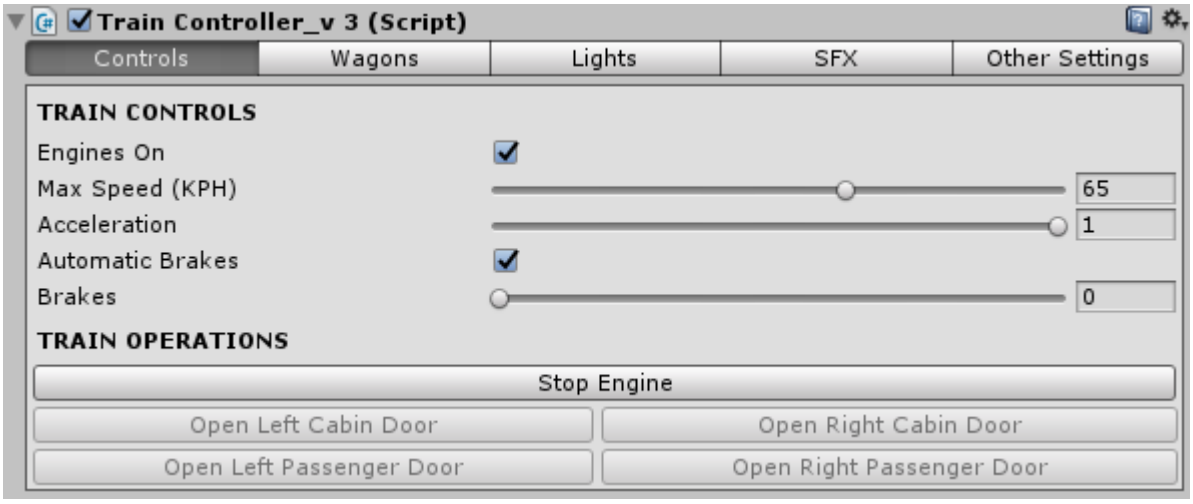
4. Train Controls

In this section you will learn how to control and monitor your train speed.
You can either have fully automated trains moving around your scene or you can have a player controlled train, either by keyboard input or Unity UI.

4.1. Automated Trains

To add an automatic train at your scene, all you need to do is to setup the train behavior by adjusting the **TrainController** script settings at the Unity Editor. The TrainController component is attached to the locomotive.

> **Note:** You can also change train settings automatically at runtime by using Control Zones..



The **Engines On** property is enabled by default, if you don't want your train to start moving automatically, you can disable it by unchecking it directly or using the button at the **Train Operations** Section.

> **Note:** Some operations are enabled only at runtime. For example, all door related operations.

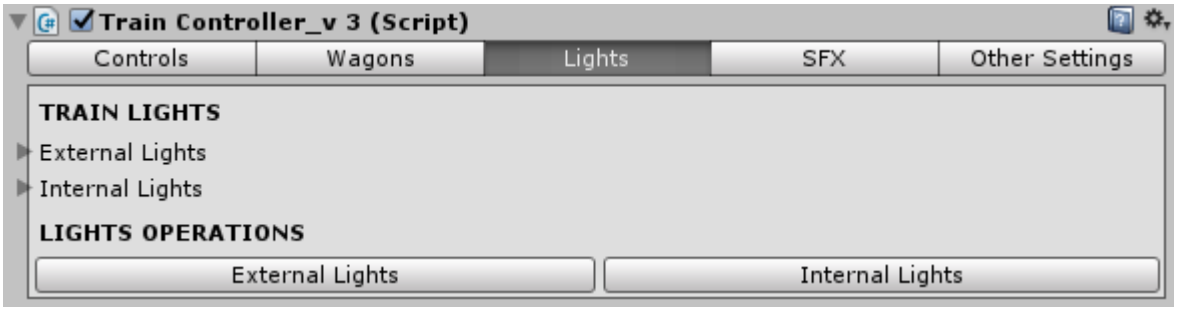
The train speed is controlled by the **Max Speed Kph** and **Acceleration** properties of the Train Controller script.

The Max speed is a range between 0 kph and 105 kph, based on real world trains average max speed values.

The acceleration property, defines train movement direction:

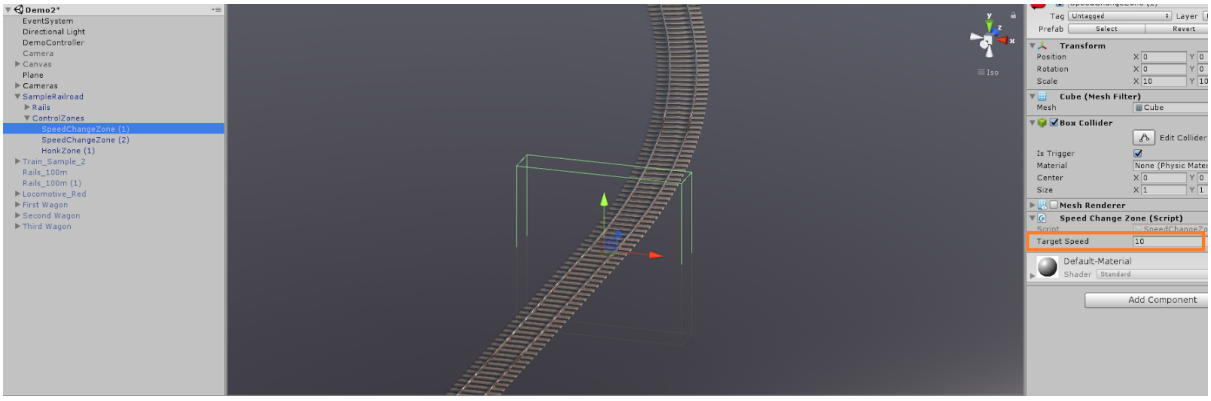
- **Acceleration 1:** Forward
- **Acceleration 0:** Stop
- **Acceleration -1:** Reverse

You can also use the **Brake** property to reduce train speed and stop the train. When the **Automatic Brakes** property is enabled, the brakes are applied automatically when the acceleration property is reduced.
You can switch you train lights on/off in the Unity Editor by using the buttons available at the **Light Operations** section. By default, internal lights are on and external lights are off.

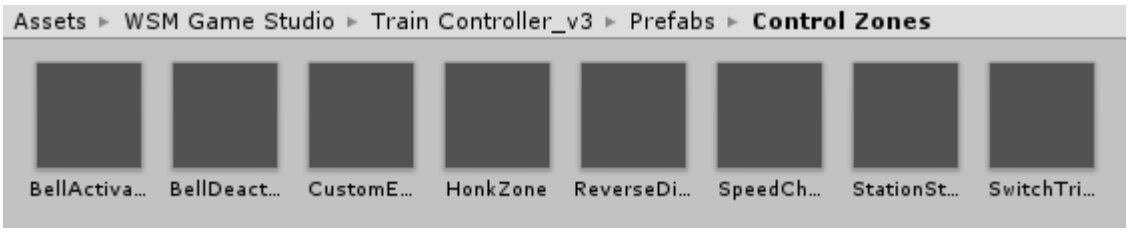


4.1.1 Control Zones

Control Zones are used to control the train behaviour on the railroad at runtime. This is very useful to automate your railroad.



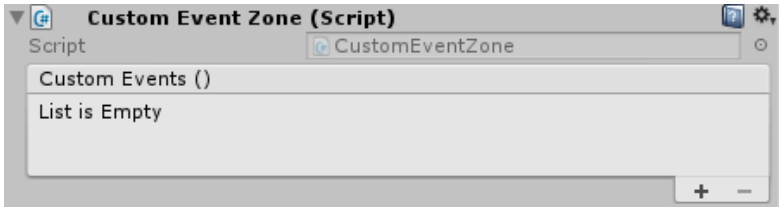
These zones are simple colliders that triggers events on the TrainController script once the train pass through them.



- **Bell Activation/Deactivation Zones:** Activates/deactivates freight train bell
- **Bell Zone:** Deactivates freight train bell
- **Honk Zone:** Activates train horn
- **Reverse Direction Zone:** Reverse train direction
- **Speed Change Zone:** Changes train max speed value
- **Station Stop Zone:** Automatic station stop (see the Stopping at the Station section)
- **Switch Trigger Zone:** Switch rails direction (see the Railroad Traffic Control section)
- **Custom Event Zone:** Trigger custom user events

> **Note:** It is recommended to use Speed Change Zones to reduce train speed before railroad curves and increase speed on straight segments. This will add more realism to your scene and increase train stability on curves.

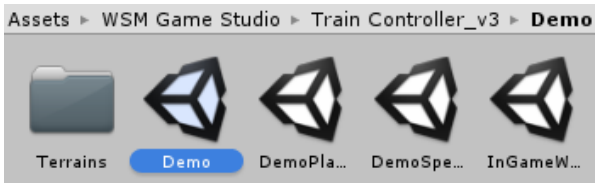
If none of the existing zones fits your project needs, you can use a **Custom Event Zone** to trigger custom events on your scene. This opens infinite possibilities on what you can do on your game.



4.2. Player Input

There are two ways of controlling a train with player input: by using the Unity UI or customizable keyboard input.

The Unity UI method, basically works by overriding the automatic train settings and calling TrainController component public methods directly. Take a look at the first **Demo** scene for sample on how to control your train using the Unity UI.



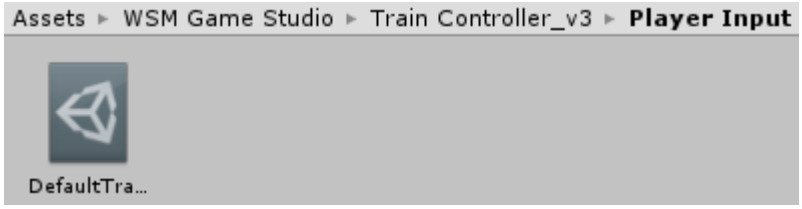
To get keyboard input, you need to enable the **Train Player Input** component on your locomotive.

Note: For Unity UI input, make sure the **Enable Player Input** property is disabled. This property is intended to be used for keyboard input only.

This component is responsible to process player input and trigger train default and custom events. They keys configuration are defined by the Input settings and can be easily customized. You can even have multiple trains, with distinct inputs being controlled at the same time.

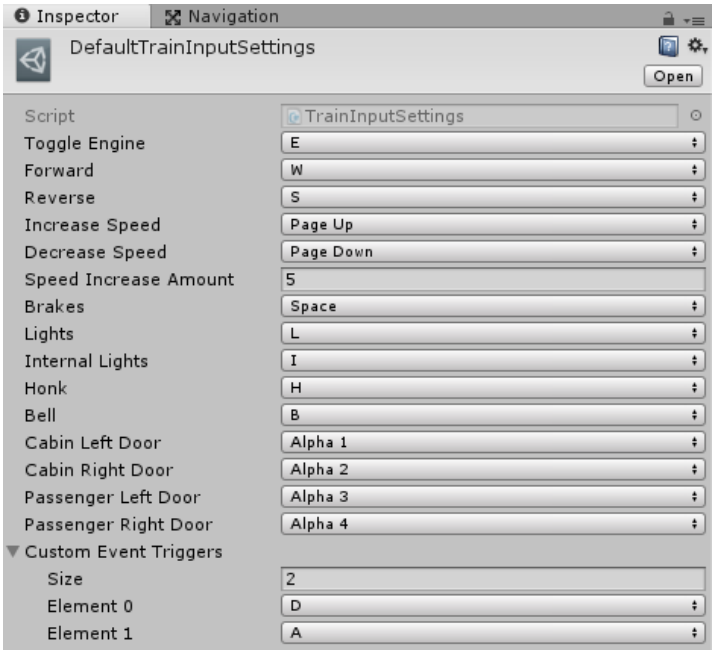
4.2.1. Custom Input Settings

The default input settings is located inside the **Player Input** folder.



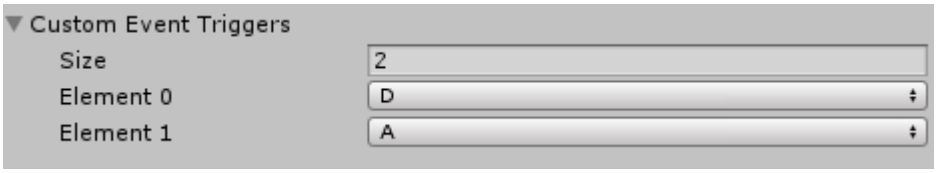
Note: Train Input Settings are stored inside Scriptable Objects, which means, you can create your own custom input settings either by duplicating the file or by using the Unity Editor creation menu:
Right Click > Create > WSM Game Studio > Train Controller > Train Input Settings

In the image below, you can see the default key input settings. You also see this in action at the **DemoPlayerInput** demo scene.

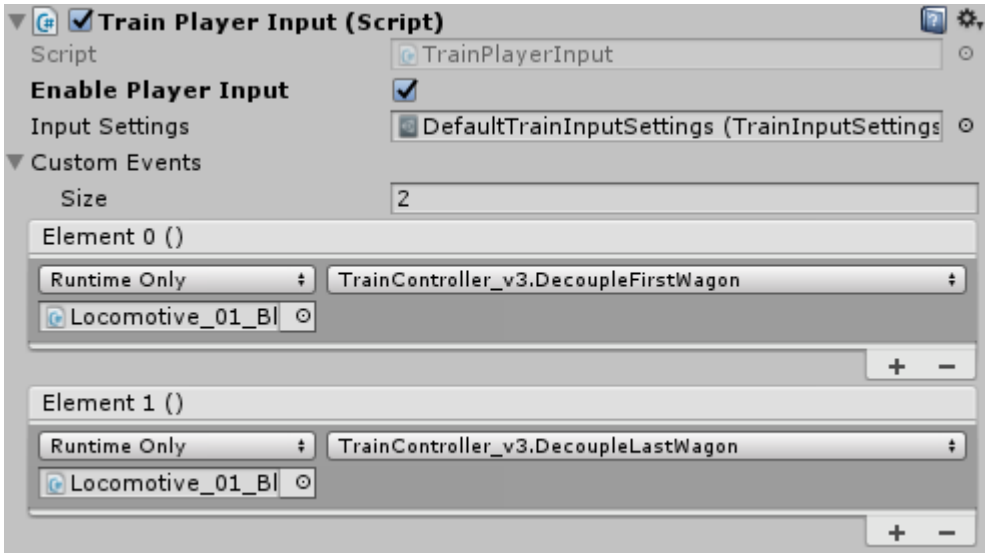


4.2.2. Custom Events

Besides the default train input, you can also create custom events triggered by player input. The key mapping for these events must be added on the **Custom Event Triggers** property of your train input settings file.



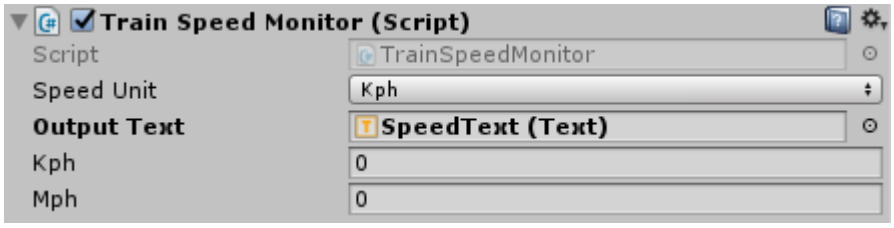
The corresponding events must be included at the **Custom Events** property of the Train Player Input component attached to your locomotive.



Note: You can have as many custom events as you wish. Just make sure to fill the **Custom Event Triggers** on your input setting file and the **Custom Events** property of the Train Player Input component attached to your locomotive.

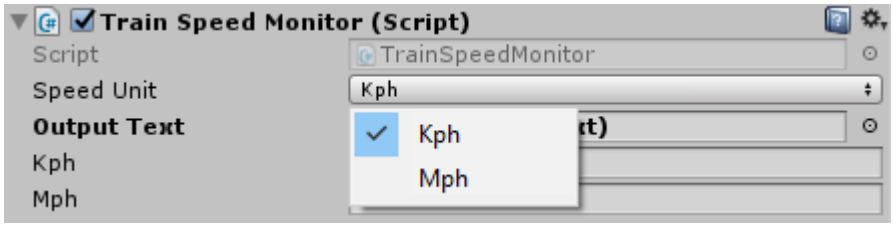
4.3. Train Speed Monitor

You can monitor train current speed, by using the **Train Speed Monitor** attached to the locomotive prefab.

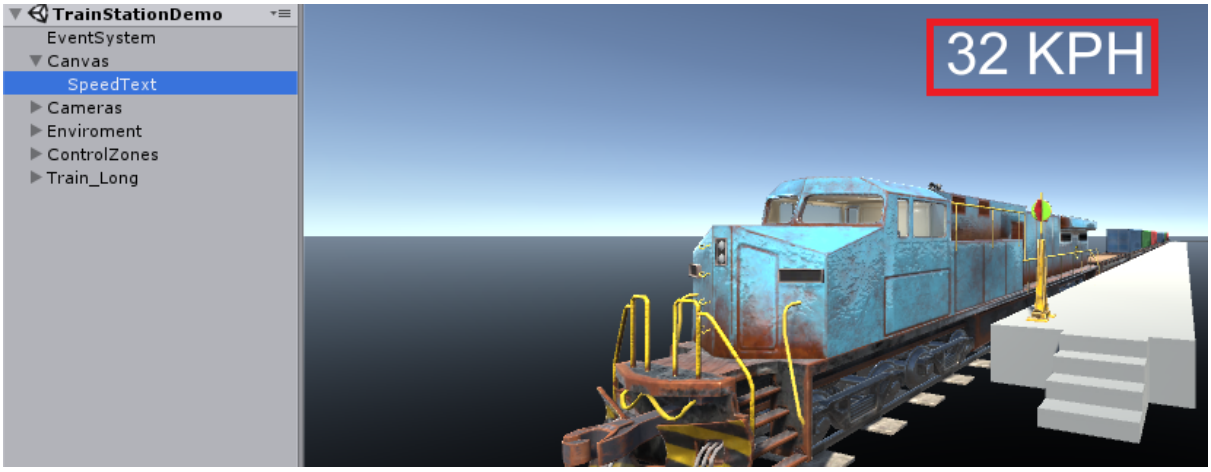


The “Kph” and “Mph” properties are updated in real time, to monitor speed by selecting the locomotive on the editor.

You can also print the current speed value to a Text field, just Drag & Drop your text to the “Output Text” property and select the speed unit you wish to print (Kph or Mph)



In all the Demo scenes included in the project, the speed is printed to the **SpeedText**, as you can see below.



5. Building a Train

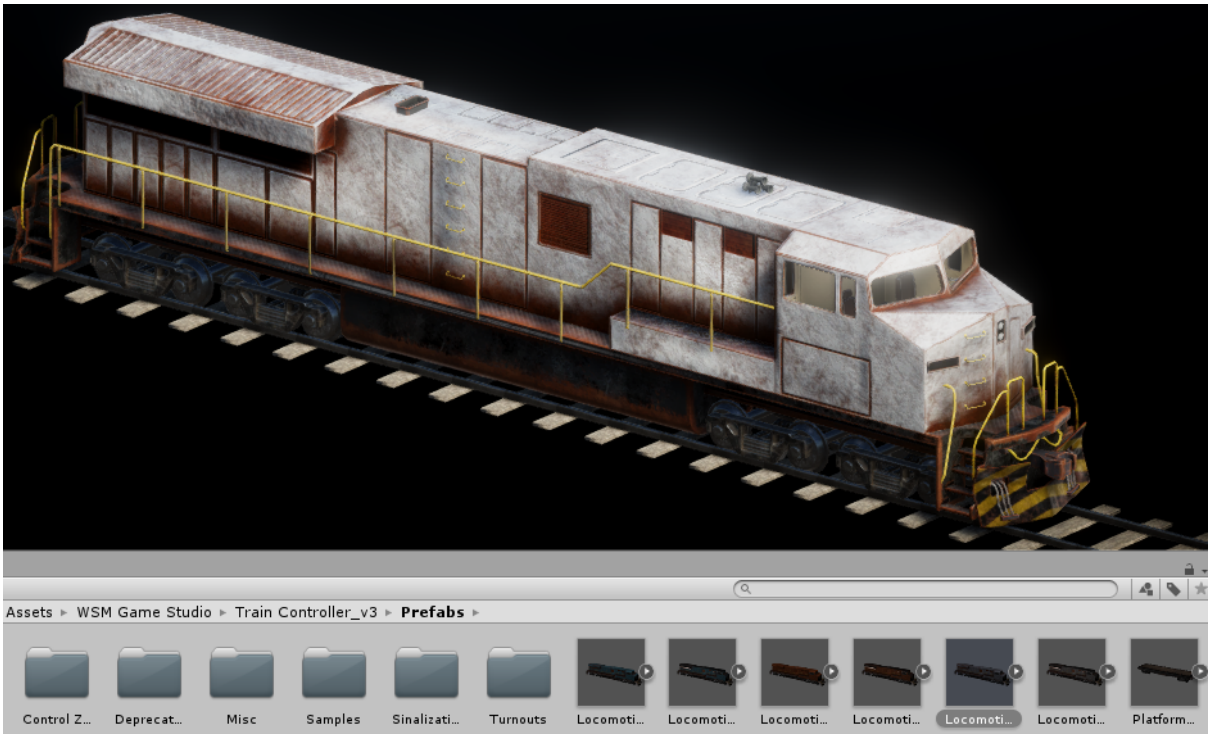
In this section you will learn how to build your custom train using the locomotive and wagons prefabs.

You can also learn this by watching this Video Tutorial instead.

5.1. Connecting Wagons (Editor)

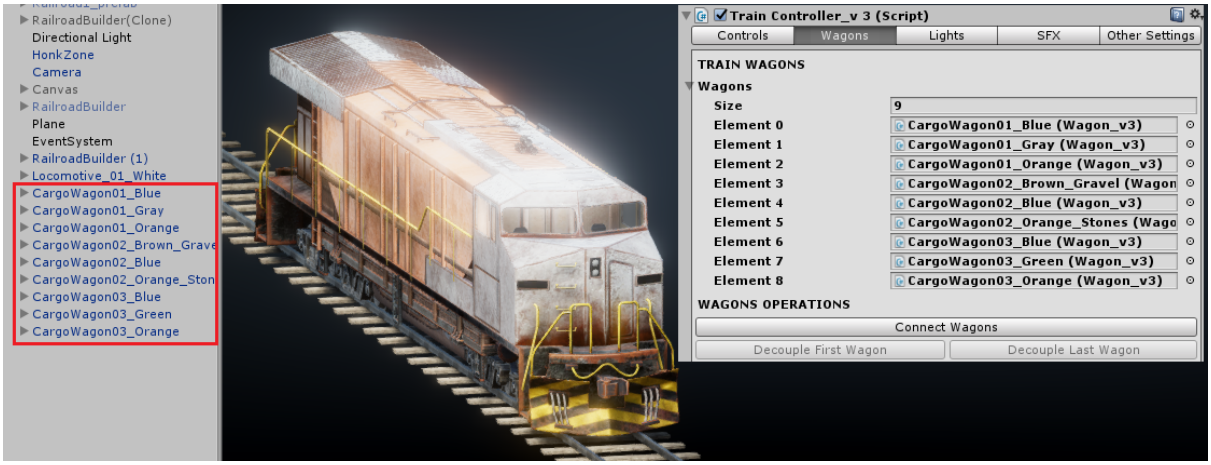
Additional wagons were used in this sample (Available Here)

To start building your train, simple drag and drop the locomotive prefab on top of a straight rail segment.



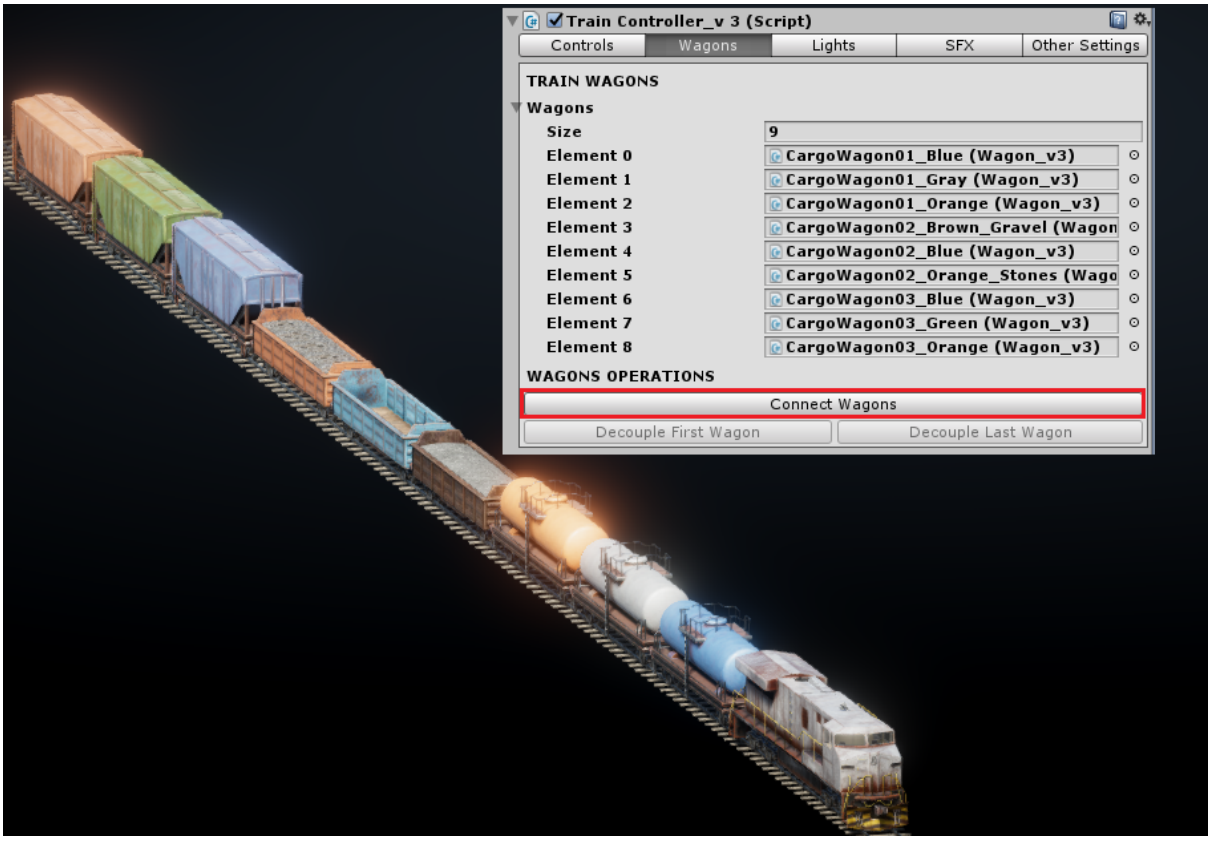
Drag & Drop wagons prefabs on your scene.

Then, select the wagons on your scene and drag & drop then on the “Wagons” list property of the “Train Controller” script attached on the locomotive prefab.



Now, click the **Connect Wagons** button at the **Wagon Operations** section.

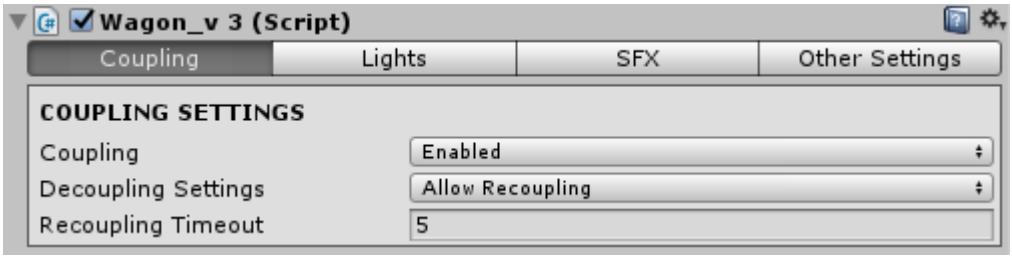
All the wagons will be repositioned automatically behind the locomotive, following the wagons list order.



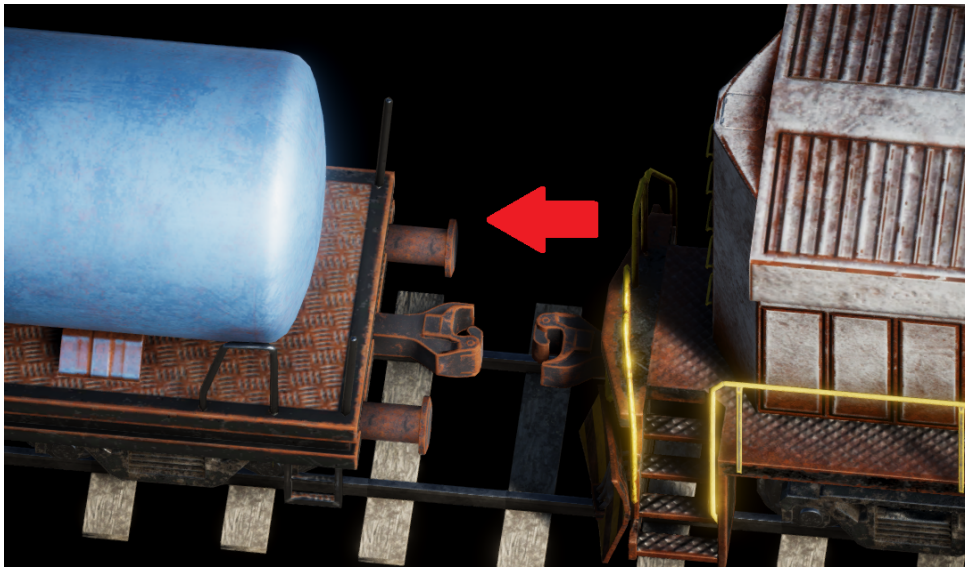
5.2. Connecting Wagons (In-Game)

Starting with version 3.3, now is possible to couple and decouple wagons in game. Take a look on the **InGameWagonConnectionDemo** demo scene.

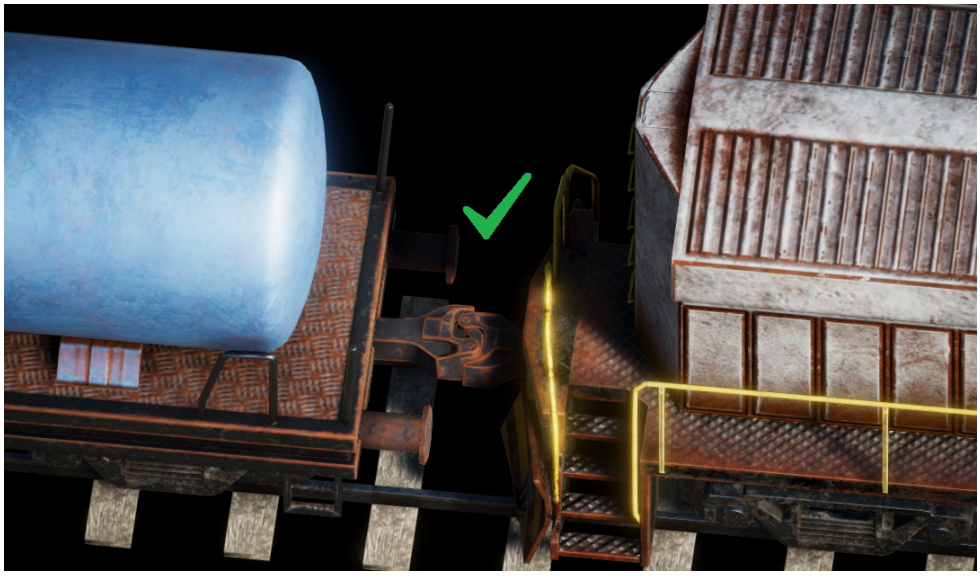
By default, in game coupling is enabled for all wagons prefabs. You check the couple settings under the **Coupling** tab in the Wagon component attached to each wagon prefab.



Here, you can enable and disable wagon coupling and configure if this wagon should be allowed to be re-coupled after being decoupled from the train for the first time.
To couple a wagon to your train in game, all you need to do is to use the reverse gear and slowly approach the wagon until you hear the connection SFX feedback.



After connecting, the couplers should look like this:

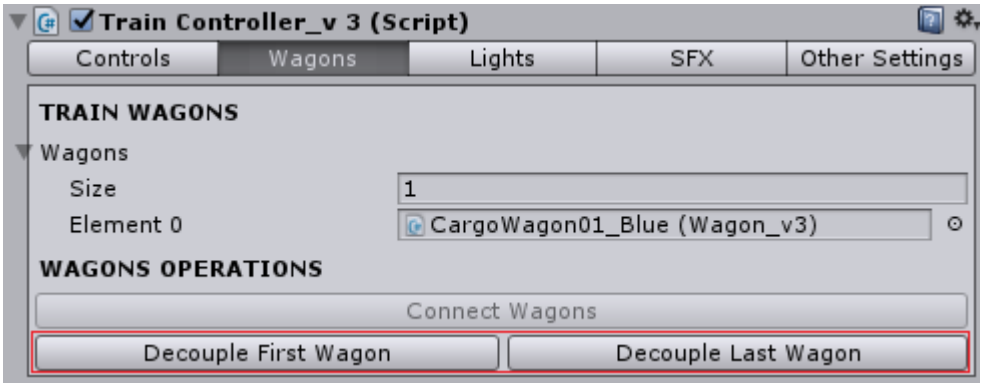


Wagons couplers are universal, so they can be connected from either side. You can also use the locomotive front coupler to connect to a wagon if you wish so.

Note: You can connect as many wagons as you wish to your train. Everytime couplers collide, they will connect, provided the coupling is enabled for that wagon.

You can also decouple wagons from your train in game. You can either decouple the first, the last or any other wagon.

While in the Unity Editor, you can test wagon decoupling the using the buttons on the **Wagons Operations** section.



If you have a train controlled by player input, using the default input settings, you can use D and A keys to decouple the first and last wagons respectively.

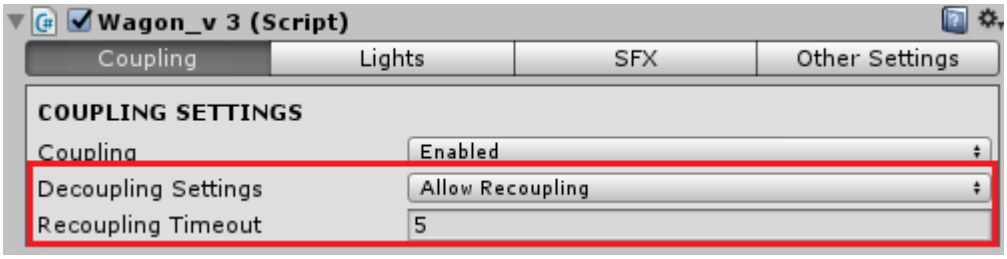
Note: The decoupling methods are configured on the **Custom Events** at the **Train Player Input** component. This is a sample on what can be done by using custom events.

You can also call the decoupling methods directly if you wish so. They are public methods located at the **TrainController** script (attached to the locomotive).

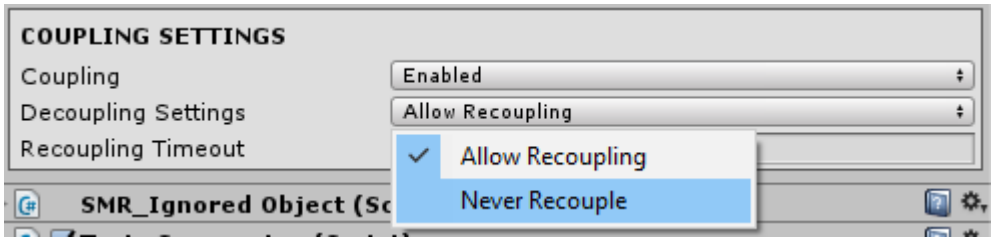
- **DecoupleLastWagon**()
- **DecoupleFirstWagon**()
- **DecoupleWagon**(int index)

Note: When decoupling wagons by **index**, keep in mind that first wagon is equivalent to index 0, following the .NET collections index logic.

After decoupling a wagon, if recoupling is enabled, you need to move away to separate the couplers before the **recoupling timeout** is elapsed (in seconds), or it will recouple the wagon again automatically.



If you don't want to recouple again for any reasons, you should change the **Decoupling Settings** to **Never Recouple**.



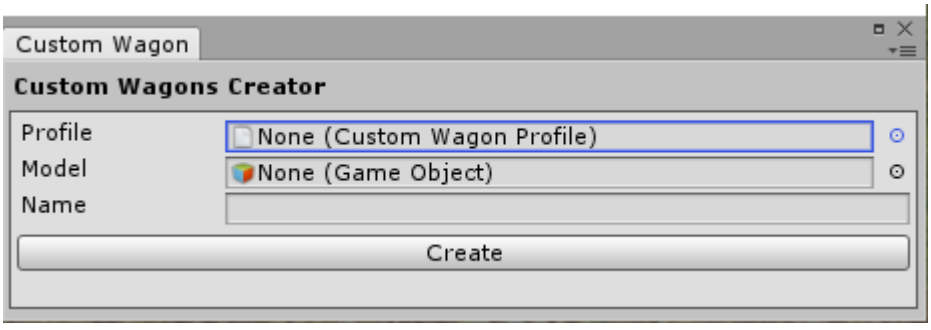
6. Custom Wagons Creator

Wagons and locomotives are complex prefabs, made specifically to be compatible with the rails generated by the Railroad Builder. Due to this complexity, it can be tricky to customize prefabs and ensure they will work properly.

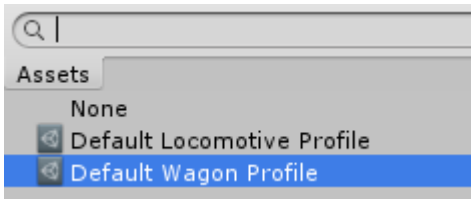
Starting with version 3.3, the custom wagon creator was introduced to increase client customization options. It can be used to create custom wagons and locomotives, based on user customizable profiles.

For example, let's assume you have a collection of wagon models you wish to use on your game. By using the custom wagon creator, you can quickly generate multiple customized wagons with your models.

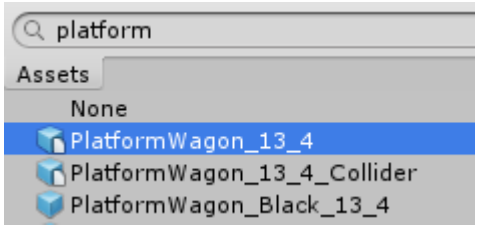
The Custom Wagon Creator is located in the Unity Editor, under **Window > WSM Game Studio > Custom Wagons Creator**



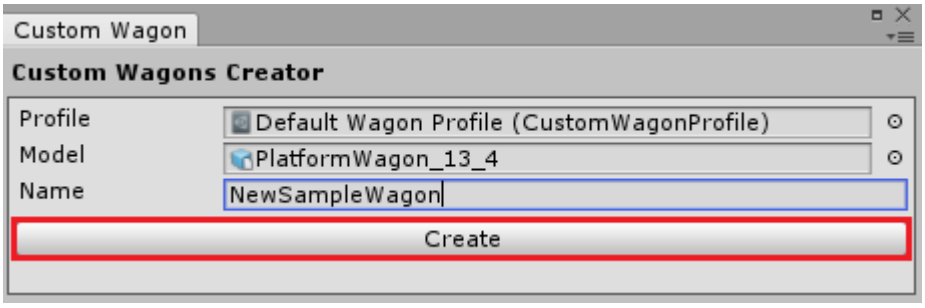
To create you custom wagon, first select the desired profile:



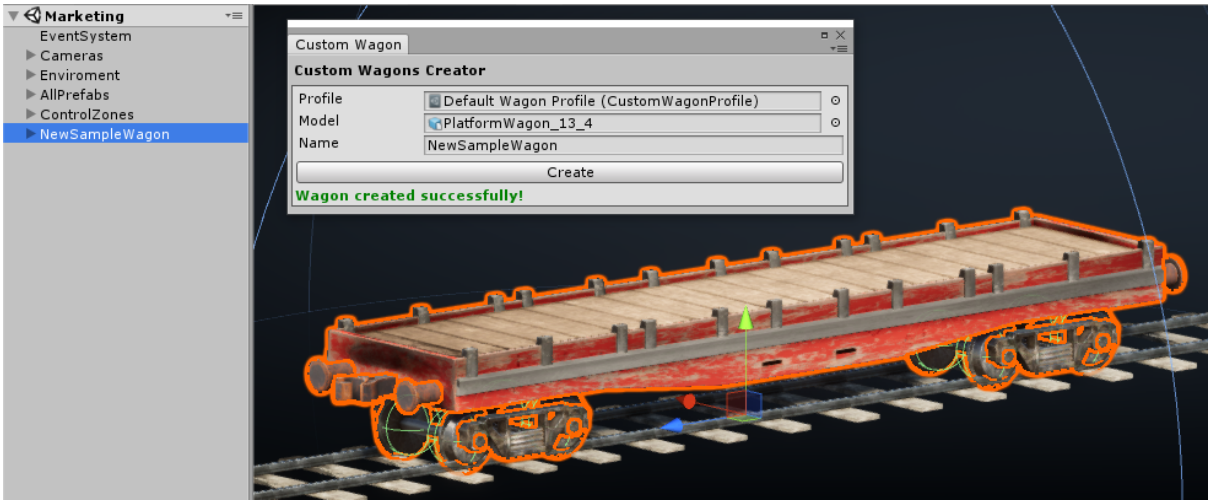
Then, select your custom model:



Select a name for the object and click on “Create”



The custom wagon will be created on your scene.



Verify if the wagon is working properly, then drag & drop it to your preferred folder to save it as a prefab.

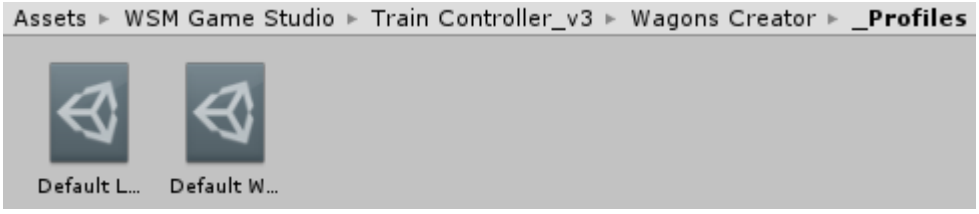
Note: You can also create custom locomotives. The type of object being created is defined on the selected profile (See the Custom Profiles section for more details)

6.1. Custom Profiles

Custom Wagons Profiles defines the type and settings of the created object.

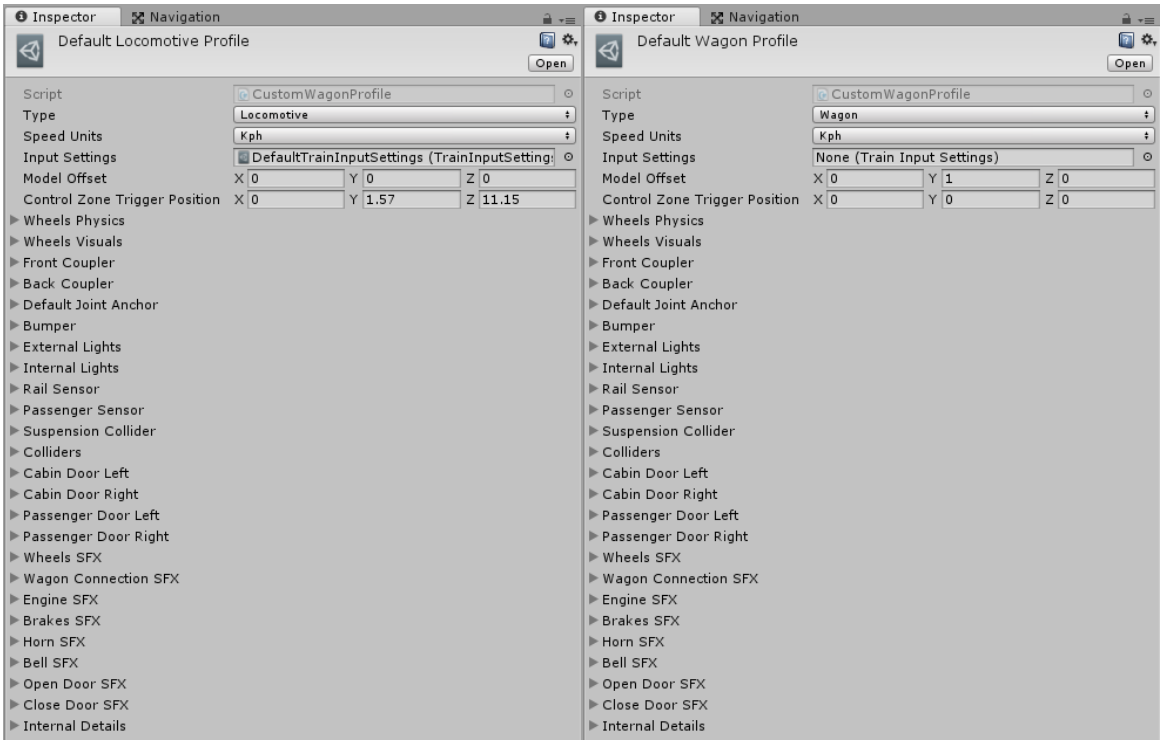
You can think of a profile like a blueprint that defines all the essential components that needs to be included in the product to ensure it will work properly.

Profiles can be used for mass creation of custom locomotives or wagons prefabs. For example, if multiple wagons models shares the same configuration, like length, wheel positions, etc... You can use the same custom wagon profile to turn them into ready to use wagon prefabs.



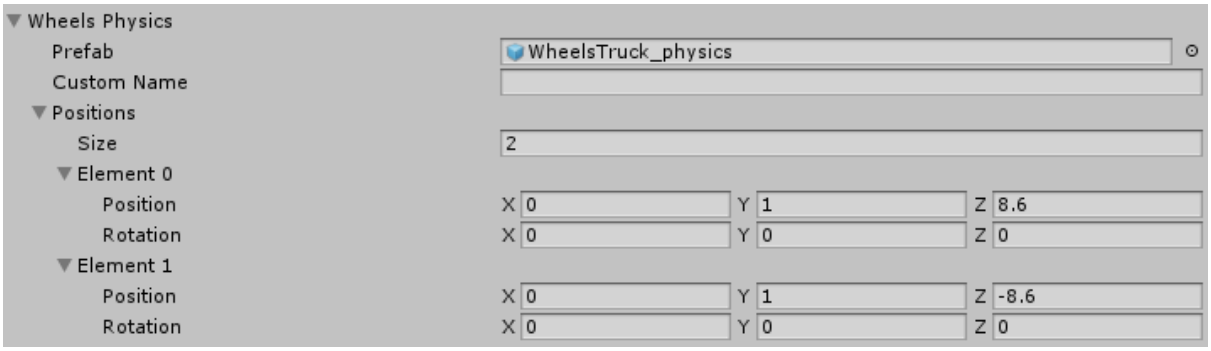
There are two sample profiles included in this package, one for creating wagons and another for locomotives.

Note: Custom Wagons Profiles are Scriptable Objects, which means, you can create your own custom profiles either by duplicating the existing files or by using the Unity Editor creation menu:
Right Click > Create > WSM Game Studio > Train Controller > Custom Wagon Profile

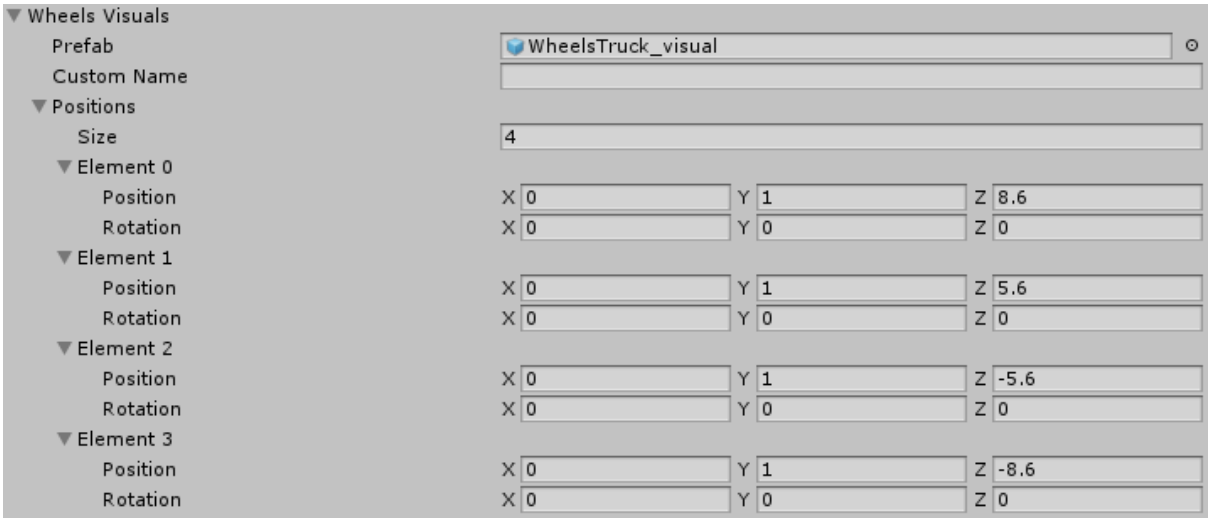


As can be seen in sample image above, profiles are very complex Scriptable Objects that contains references to all the Parts that will compose your custom wagon. It also defines each type of object will be created (locomotive or wagon).

For example, the **Wheels Physics** section, holds a reference to the prefab responsible for simulating train wheels physics, and the **Positions** array defines how many and where the instances should be spawned.



The **Wheels Visuals** section, holds a reference to the prefab responsible for simulating the wheels turning animation, how many and where the instances should be spawned.



Note: As you can see in the sample image below, there are four visual wheels, but only two physics wheels on the locomotive profile. That's because there is no need for more than two physics wheels to keep the locomotive on track, even though the locomotive has four sets of wheels.

6.2. Wagons/locomotive Parts

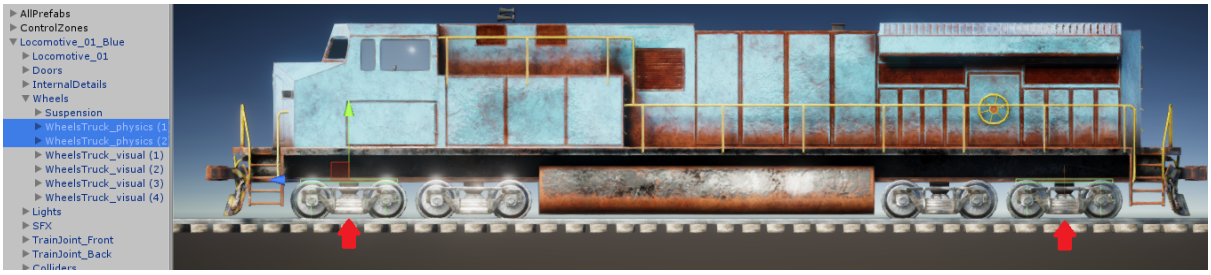
Parts are the essential components needed to create working locomotives and wagons. The default parts prefabs are stored inside the **Wagons Creator > Parts** folder.

There are many different parts, but most of them are self explanatory. In this section, you will know more about the mandatory parts and how to set them properly on your custom profile.

The mandatory parts are:

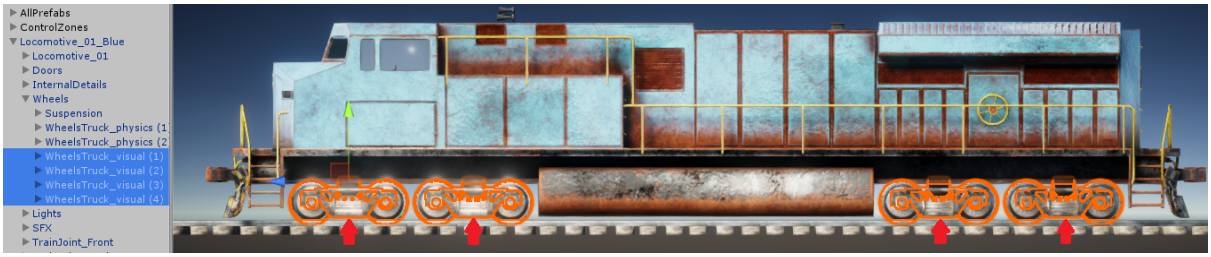
- Physical Wheels
- Visual Wheels
- Front and Back Couplers
- Default Joint Anchor **(wagon only)**
- Rail Sensors
- Suspension Colliders
- Colliders
- Wheels SFX
- Wagon Connection SFX **(wagon only)**
- Engine SFX **(locomotive only)**
- Brakes SFX **(locomotive only)**
- Horn SFX **(locomotive only)**
- Bell SFX **(locomotive only)**

Physical Wheels are components responsible for physically interact with the rails and keep the train on the track. It is recommended to always use only two physical wheels, one at each end (front and rear). Two is the minimum required and by keeping that way you ensure you won't have unnecessary rail friction.



Note: Starting from v3.3, the physical wheels components don't have wheels models being rendered on them anymore. Visual and physics elements were separated for better results.

Visual Wheels are components responsible for rendering the train wheels and the wheels rotation animation. You can have as many visual wheels as you want.

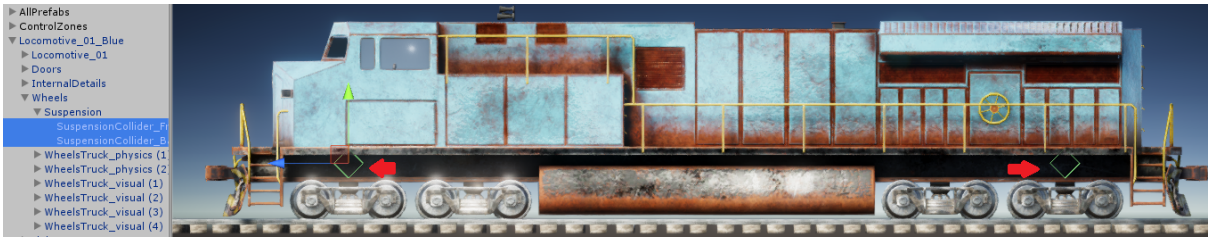


The **Front and Back Couplers** are components responsible for connecting wagons, either on the Unity Editor or in-game (Take a look on the Building a Train section for mode details).

The **Default Joint Anchor** is a component responsible for stabilize wagons that are not connected to any train. This anchor was created as a workaround to fix a PhysX issue with disconnected joints.

Rail Sensors are components responsible to verify if the train is grounded and on rails. That are two of them (left and right), they must be located on the lower part of the first set of wheels.

Suspension Colliders are auxiliary components used to maintain train stability on terrain elevations. They must be located above the physics wheels components.



Colliders are prefabs containing the default Unity Colliders. If you intend on creating similar wagons that share common colliders, you can save them as prefabs and set them at your custom profile.

The **SFX** components are prefabs that contains Audio Sources with train related SFXs.

Note: Default parts are intended to be used as a starting point for creating working locomotives and wagons. Therefore, it is **NOT** recommended to edit them directly.

If you wish to take your customization to the next level, by changing wheels models or using different SFXs for example, you can create custom parts, by duplicating the existing parts prefabs and customizing them using the default Unity's prefab editing workflow.

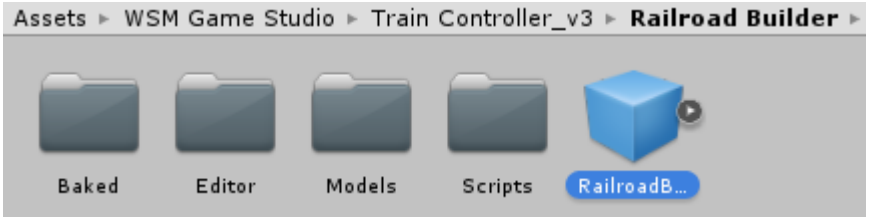
7. Building a Railroad

In this section you will learn how to build your custom railroad, using the Railroad Builder prefab.

You can also learn this by watching this Video Tutorial instead.

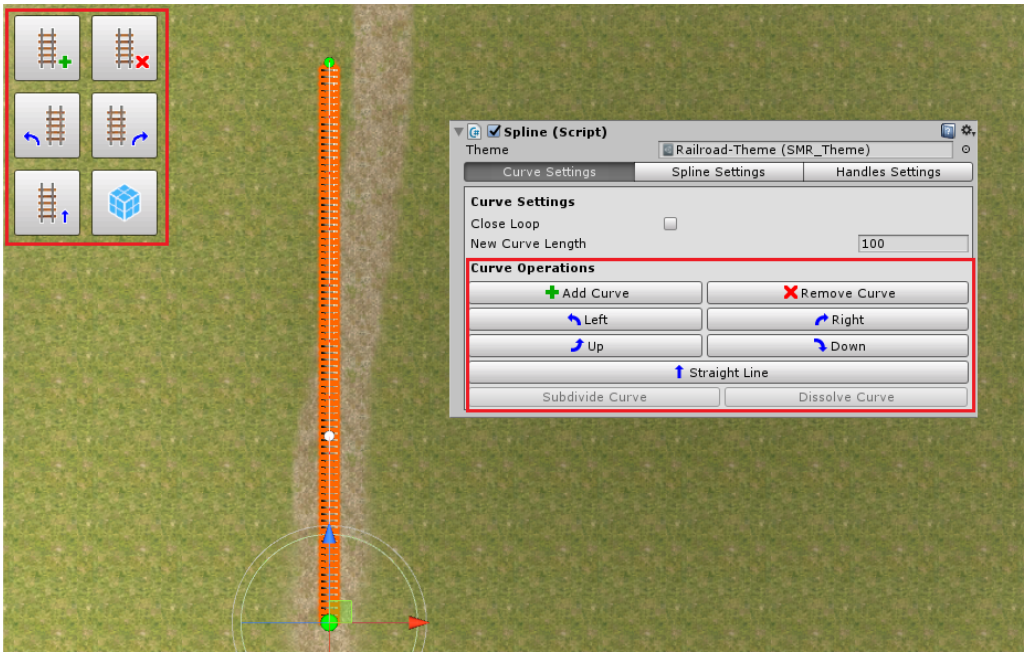
7.1. Railroad Builder

The **Railroad Builder** prefab, allows you to build a railroad using by adjusting bezier curves on the Unity Editor.



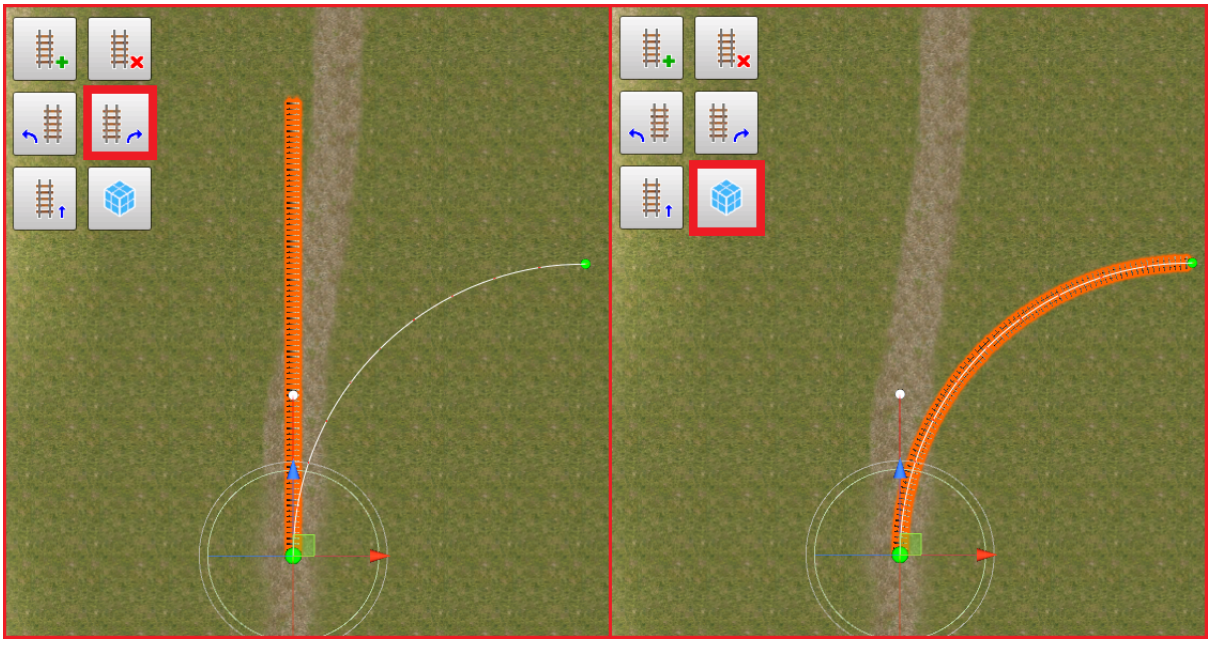
IMPORTANT: Do **NOT** apply modifications to the **Railroad Builder** prefab. This prefab is intended to be used as a starting point. If you wish to save your edited Railroad, you can save it on a new prefab or you can use the **Mesh Baker** to export a static and performance friendly railroad prefab.

Drag & Drop the **Railroad Builder** prefab into your scene to start building your railroad. It will spawn a straight railroad segment along a spline delimited by control points. The green control points, marks the start and end of each railroad segment, the white ones are auxiliary control points.

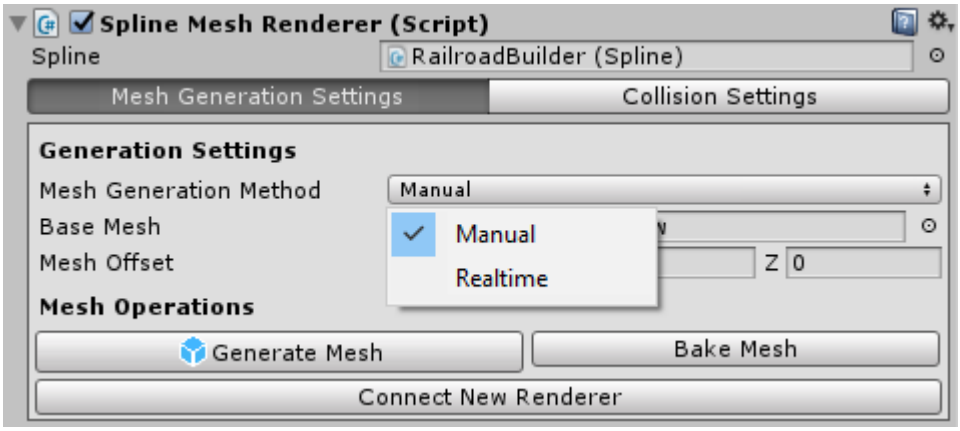


By clicking on a control point, it's position and rotation handles will appear on the editor. Change the position of the control points to curve the rail segment manually. However, there is also curve operations you can use to manipulate your curve faster. This can be seen in the sample image above in the top left corner of the screen, and also on the **Curve Operations** section of the **Spline** script.

For example, if you want to curve the rail segment to the right, you can use the turn right operation. To do this, click on the **Turn Right** button to curve your spline, then click on the **Generate Mesh** button to update your railroad.

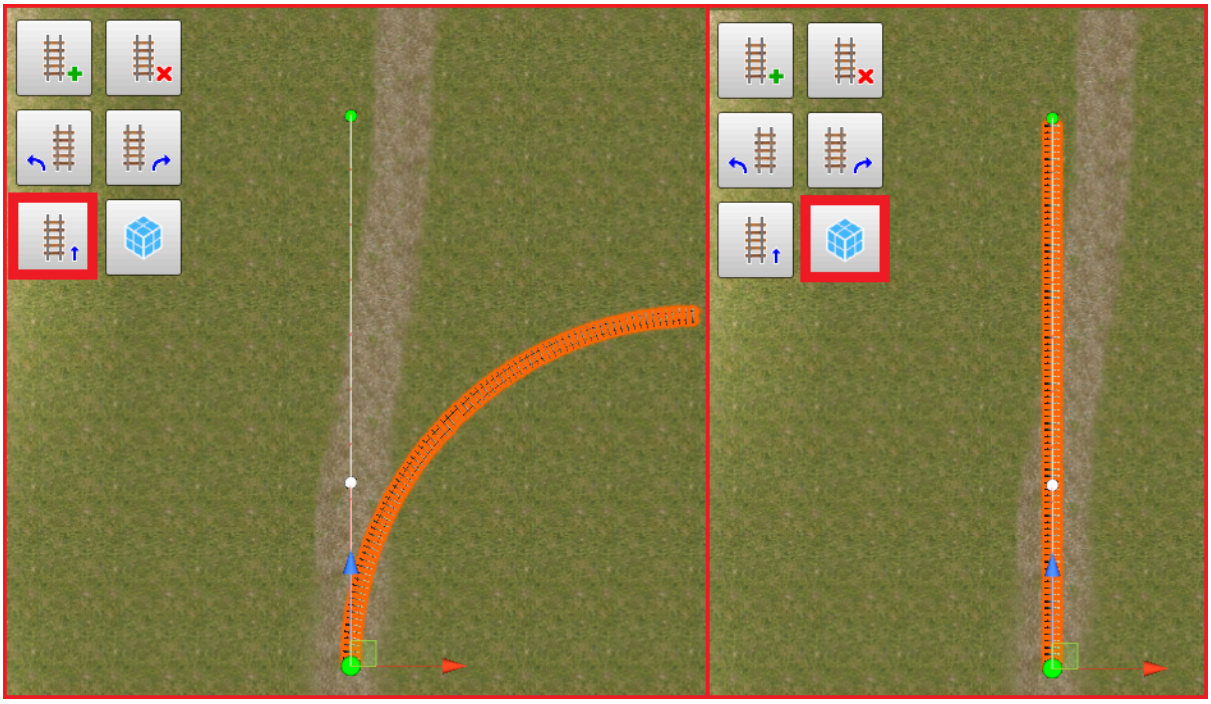


In this case, you need to manually click on Generate Mesh after curving the spline, because **Mesh Generation Method** is set to manual by default. The **manual mesh generation** allows you to control when the mesh will be generated, therefore it is performance friendly for the Unity Editor.

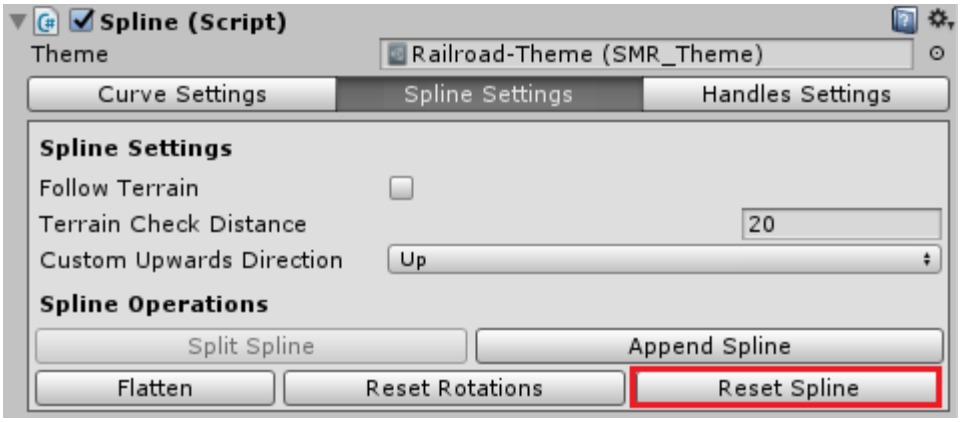


If you want to apply all spline changes to the mesh immediately, then you can use the **realtime mesh generation** option instead. However, keep in mind that this option is performance heavy and may slow-down the Unity Editor when working with long railroads.

To reset your curve segment to a straight segment, click on the straight segment button.



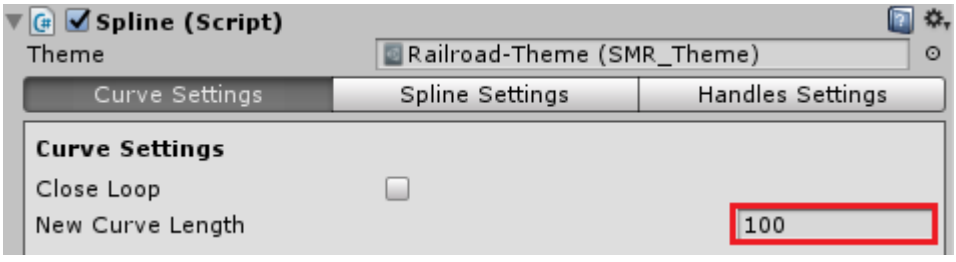
At this point, you can also do this, by using the **Reset Spline** button. However, these operations have different effects on your railroad. The straight segment only affects the last rail segment of your railroad, projecting a straight line starting on the direction of the previous curve. The Reset Spline will reset your railroad to its initial form: one single straight rail segment.



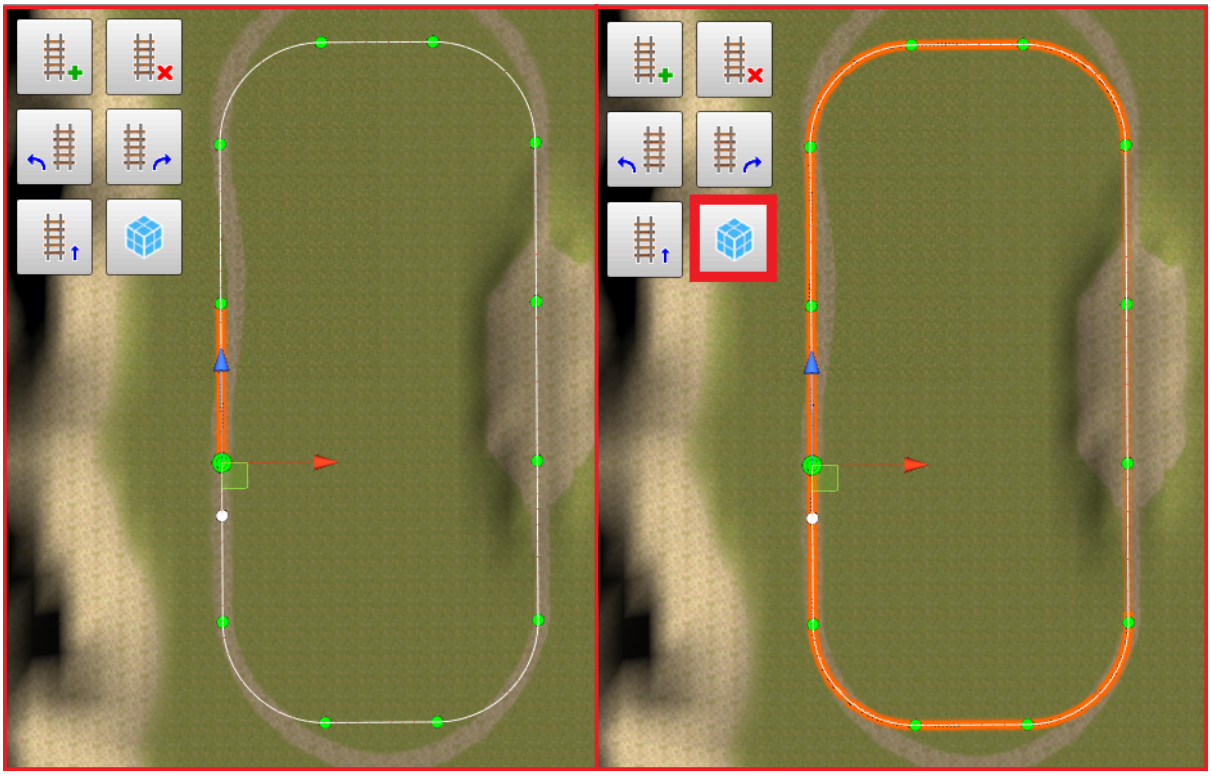
A railroad is composed by multiple segments connected to each other along a spline. To add new railroad segment you need to use the **Add Curve** button.

Note: As can be seen in the sample image above, you can also use the **Remove Curve** button to remove the last railroad segment.

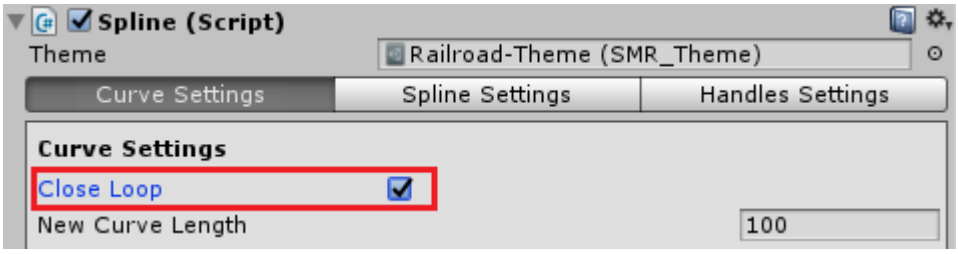
The length of the newly created railroad segment is defined by the **New Curve Length** property.



By combining these operations, and eventually adjusting the curve length, you can easily create your railroad very fast.

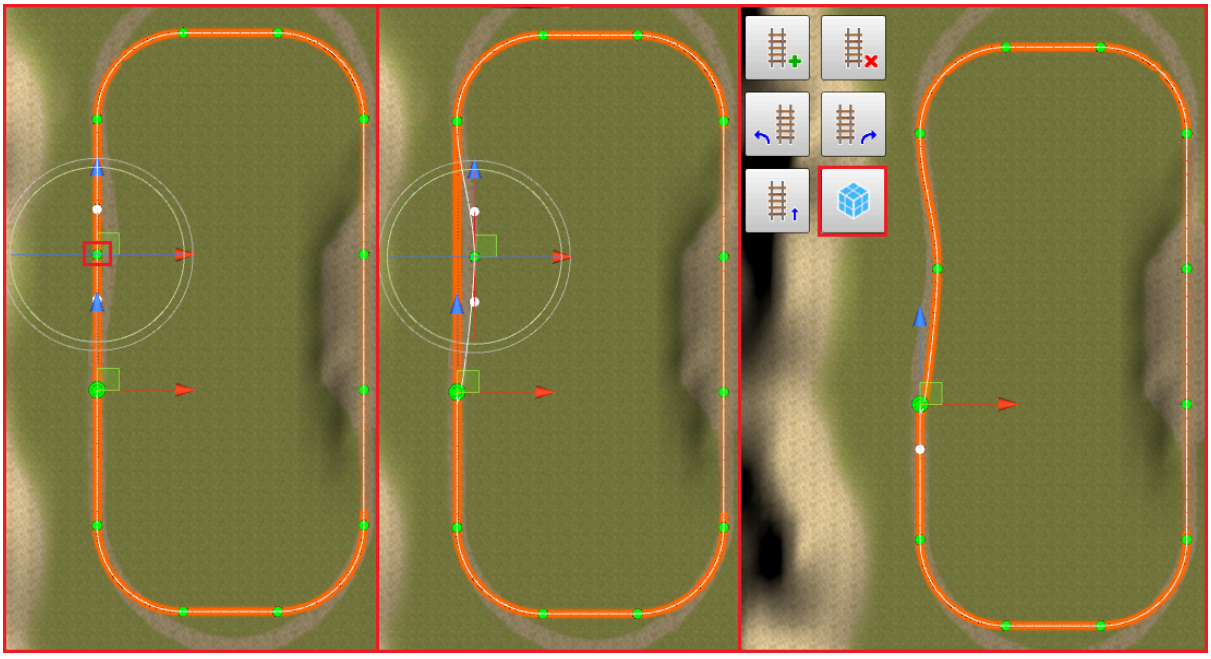


If you are building a closed railroad, after building your railroad, make sure to enable the **Close Loop** property, to ensure the rails will connect correctly at the closing point.

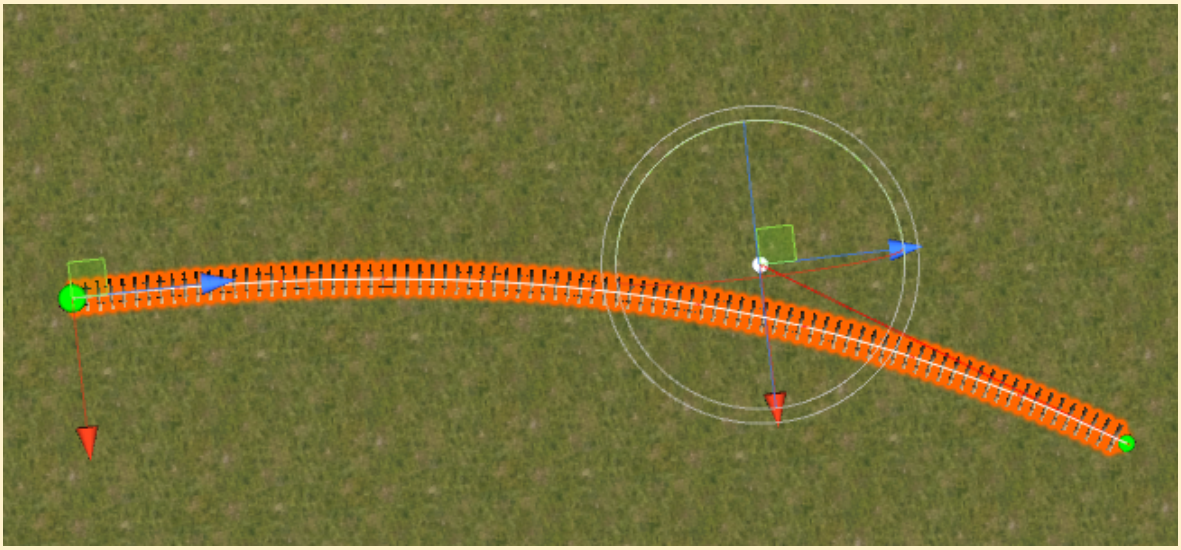


Note: When using manual mesh generation is recommended to regenerate your mesh after enabling the close loop property, just to make sure.

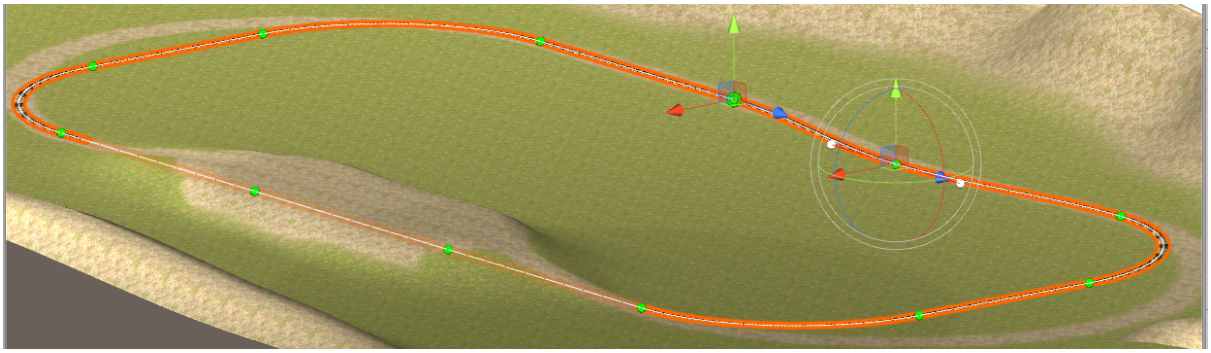
If you want to customize the curvature of the rails, you can also manually adjust any curve segment by clicking on the control points (green and white spheres). By clicking on a control point, its position and rotation handles will appear, use the handles to adjust its position and deform your railroad as you wish.



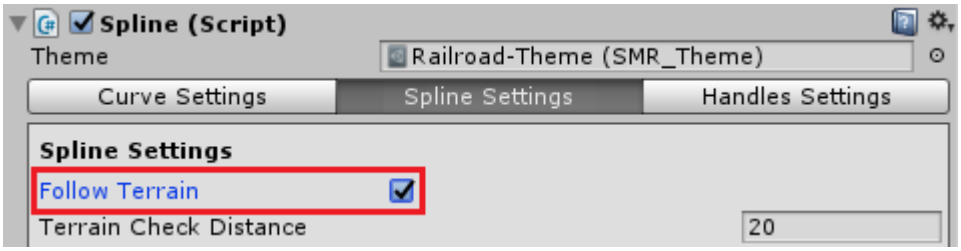
Note: By manipulating the control points you can also create curves on any angles you want. Which means, you are not restricted to the default 45º curves created b the curve operations.



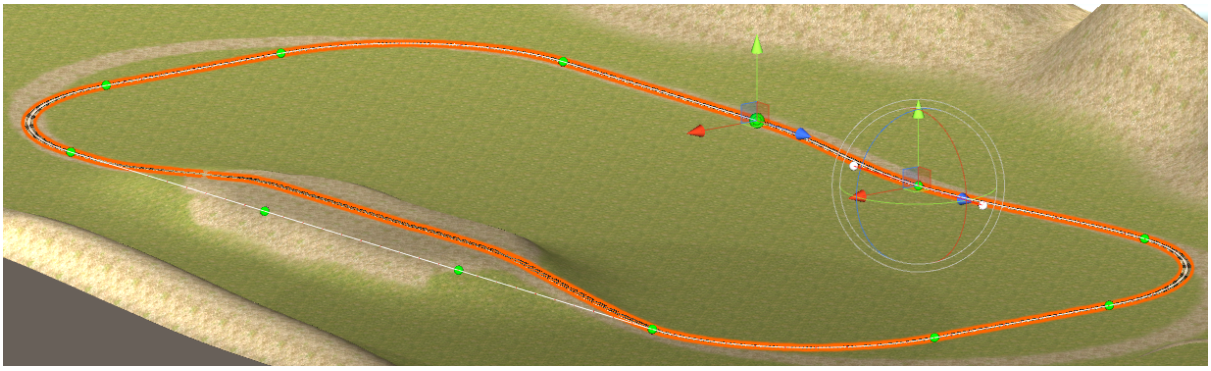
The curve operations, are very useful for creating a flat outline of your railroad, as can be seen in the sample image below.



When working with terrains, you need to enable the **Follow Terrain** property, to ensure your railroad adjust itself properly to the terrain elevations.



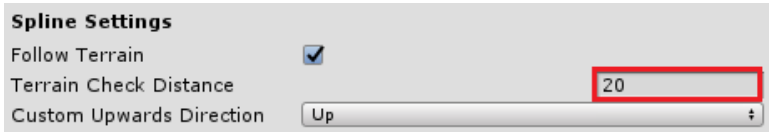
When enabled, the generated mesh is project on the terrain, even though the guide spline is still flat. This keeps the spline editing a very simple task, since you only need to build a horizontal outline of where the railroad should be and the Follow Terrain feature take care of the vertical adjustment for you.



Note: When using manual mesh generation you need to regenerate your mesh after enabling the follow terrain property.

The Follow Terrain is a very powerful feature, however, when working with very long railroads, it can become performance heavy for the Unity Editor, since it needs to check the exact position on space that the spline vertically intersects the terrain for every new segment created.

Since the vertical intersection can be located on an infinite number of spatial points below or above the spline, to avoid performance issues this feature is limited by the **Terrain Check Distance** property.



The Terrain Check Distance property defines how far from the spline the Follow Terrain feature should look for the terrain intersection.

NOTE: By default the “Terrain Check Distance” is set to 20 meters, which means it will look for the terrain 20 meters downwards and 20 meters upwards, covering a vertical range of 40 meters.

However, if your terrain height range is greater than 40 meters, you may need to adjust this value if, and only if, your spline projection does not reach a part of your terrain.

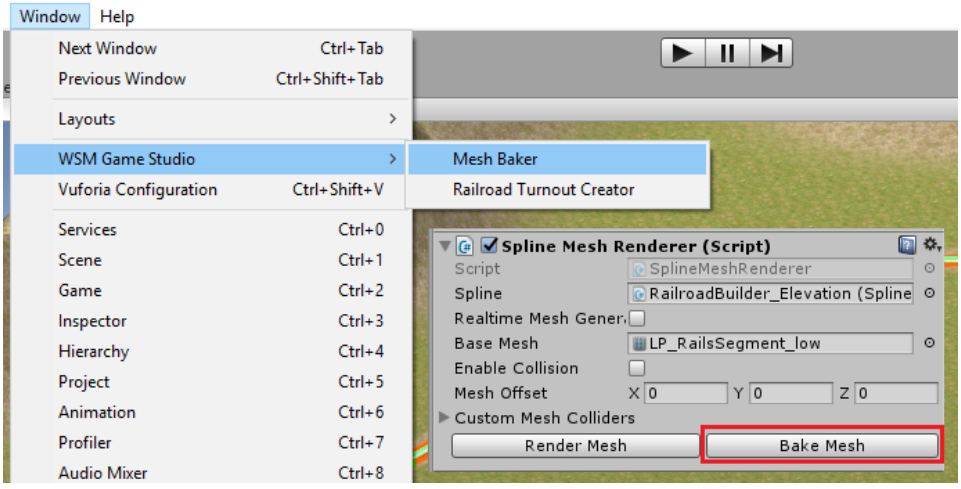
Alternatively, you can also roughly adjust the spline to bring it closer to your terrain until it reaches the check range. This alternative is performance friendly and can be done quite quickly.

Adjust the railroad as you like. After finishing the railroad creation, you can keep it as a dynamic railroad (keep Railroad Builder on scene) or you can bake the railroad as a prefab.

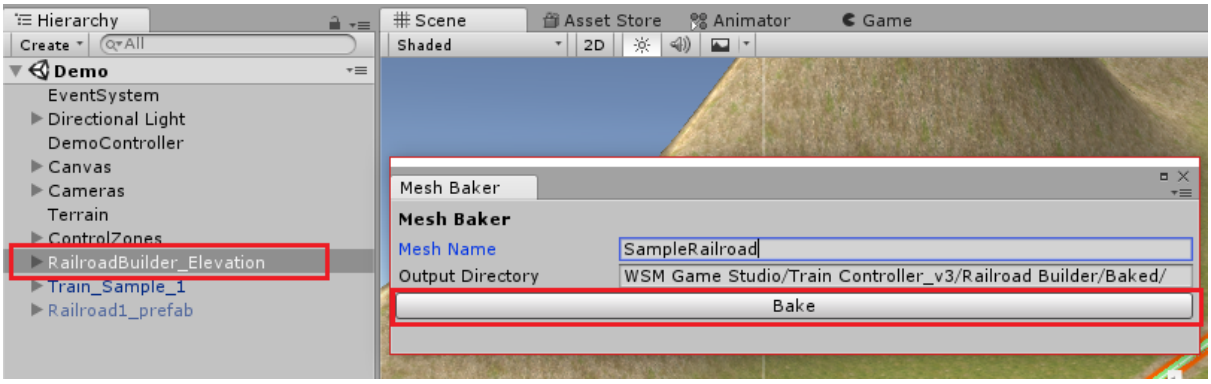
7.2. Railroad Prefab Export (Mesh Baker)

Railroads or rail segments created using the Railroad Builder can be exported as prefabs to increase performance.

Select the Railroad Builder object in your scene and click the **Bake Mesh** button or the **Mesh Baker** menu to open the **Mesh Baker Window**.

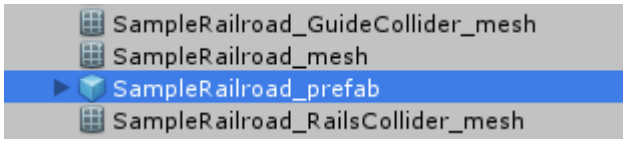


Choose a name and output directory for the prefab and with the **Railroad Builder** object still selected on your scene, click the **Bake** button.



If done correctly, four files should be created on the selected output folder.

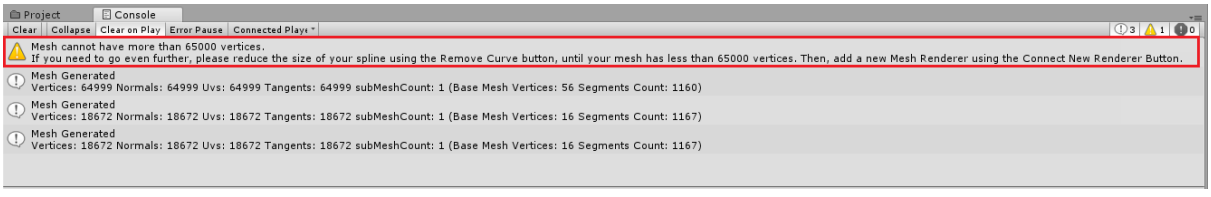
The prefab object is ready to use, just drag & drop it on your scene and disable Railroad Builder object used it to create it.



After replacing Railroad Builder by the prefab on your scene, it is recommended to keep the Railroad Builder as a disabled object on your scene, instead of deleting, in case you may want to make any adjustments later.

7.3. Building Super Long Railroads

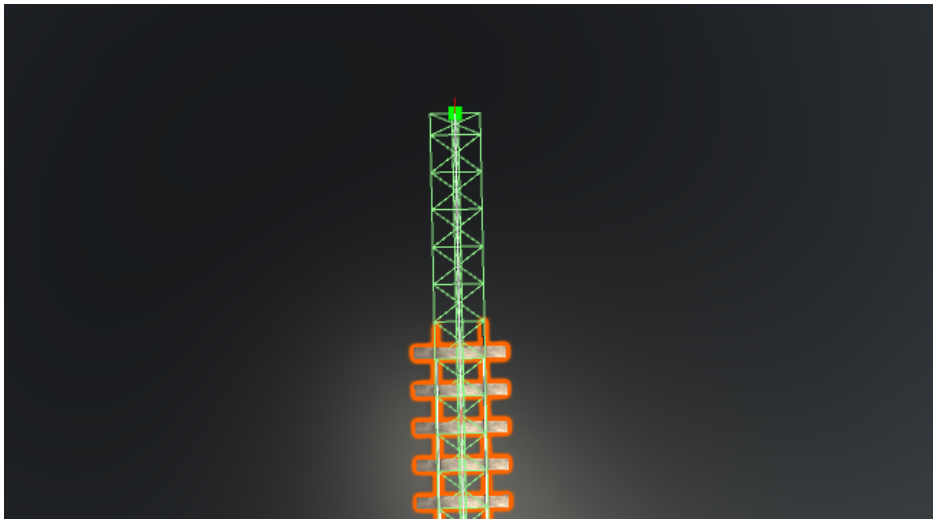
The railroad builder works by rendering the rails as a single mesh deformed along a spline. This technique allows custom rails with any curvature angle. Although, if you intend to build super long railroads, for open world games for example, eventually you may see a warning like this:



Don't worry, this is normal and **will not** ruin your plans of building a super long railroad. Unity has a limit of 65000 vertices per mesh. This limit exists to avoid performance issues when rendering large meshes. This is limit is also, very useful to apply Occlusion Culling on your railroad and increase your game performance.

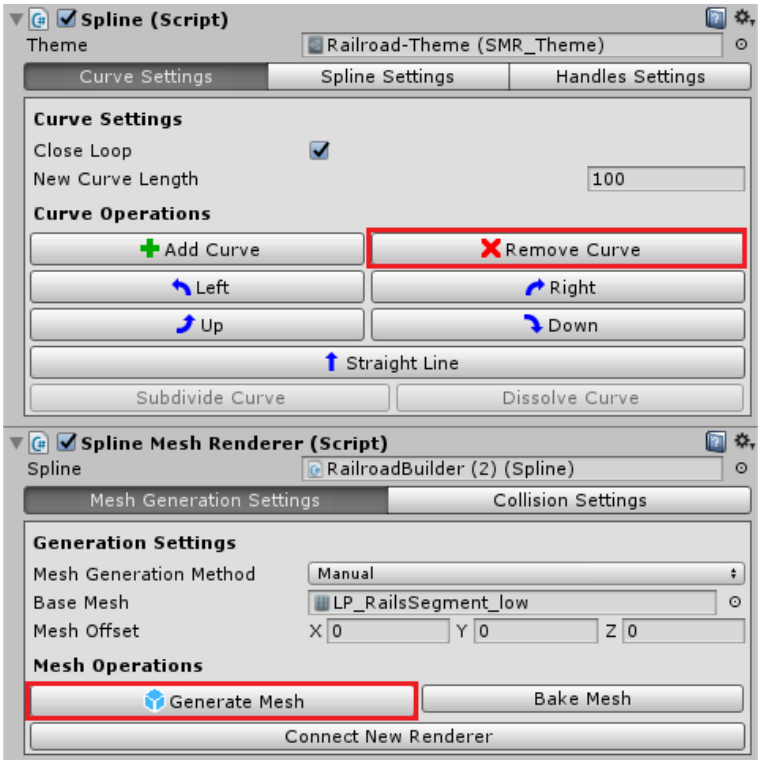
If you see that warning, just follow the instructions below to keep building your super long railroad.

With the Railroad Builder object selected, go to the end of your railroad and check if the rails matches the last control point. It probably don't match, like in the sample image below.



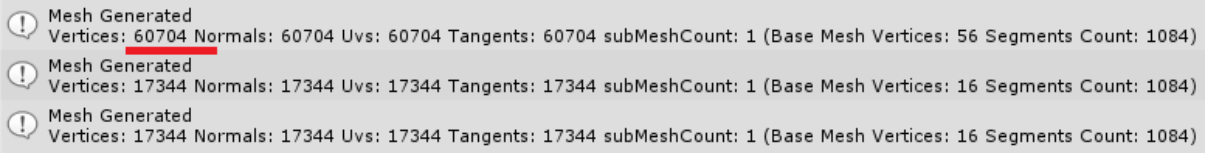
To fix that, you need to use the **Remove Curve** button to reduce your Railroad Until the rails mesh perfectly matches the last Control point position.

Click the **Remove Curve** button, then click the **Render Mesh** button and check the console window, to see if a new warning was created.

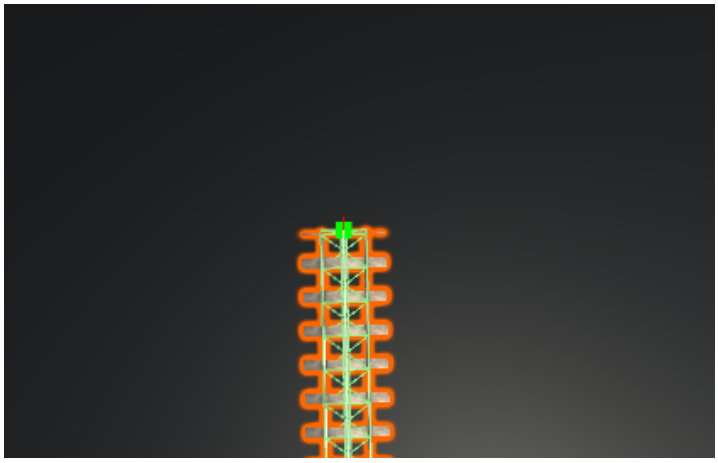


NOTE: Realtime Mesh Generation is disabled automatically when max number of vertices is greater the 65000. This is the default behaviour to increase the editor performance while building your railroad, since processing large mesh data in realtime is costly. You can re-enable the Realtime Mesh Generation manually after reducing the rails, but for very long railroads it's recommended to use the Generate Mesh button to update the mesh after adjusting your splines curves.

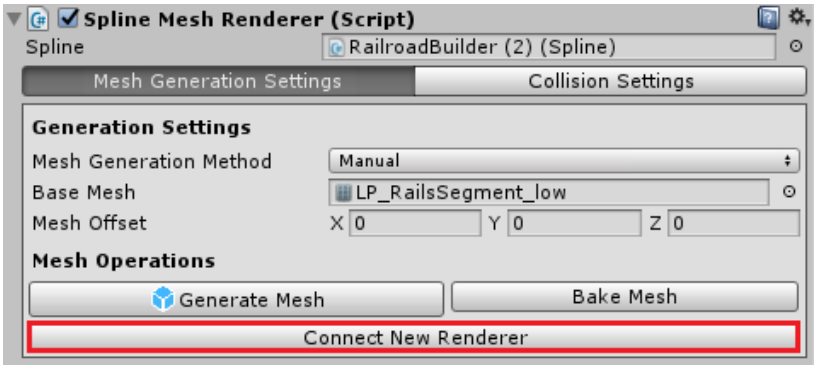
If no new warning was printed on the console after clicking on the "Render Mesh" button, you will have 3 new **Mesh Generated** outputs, like below. The first one, is relative to the rails mesh, and will be below 65000 vertices (The other two are relative to the mesh colliders)



Check the end of the railroad again. If no warning was generated, then the mesh will match the last control point perfectly. If not, then repeat the process of removing the last curve until it matches.



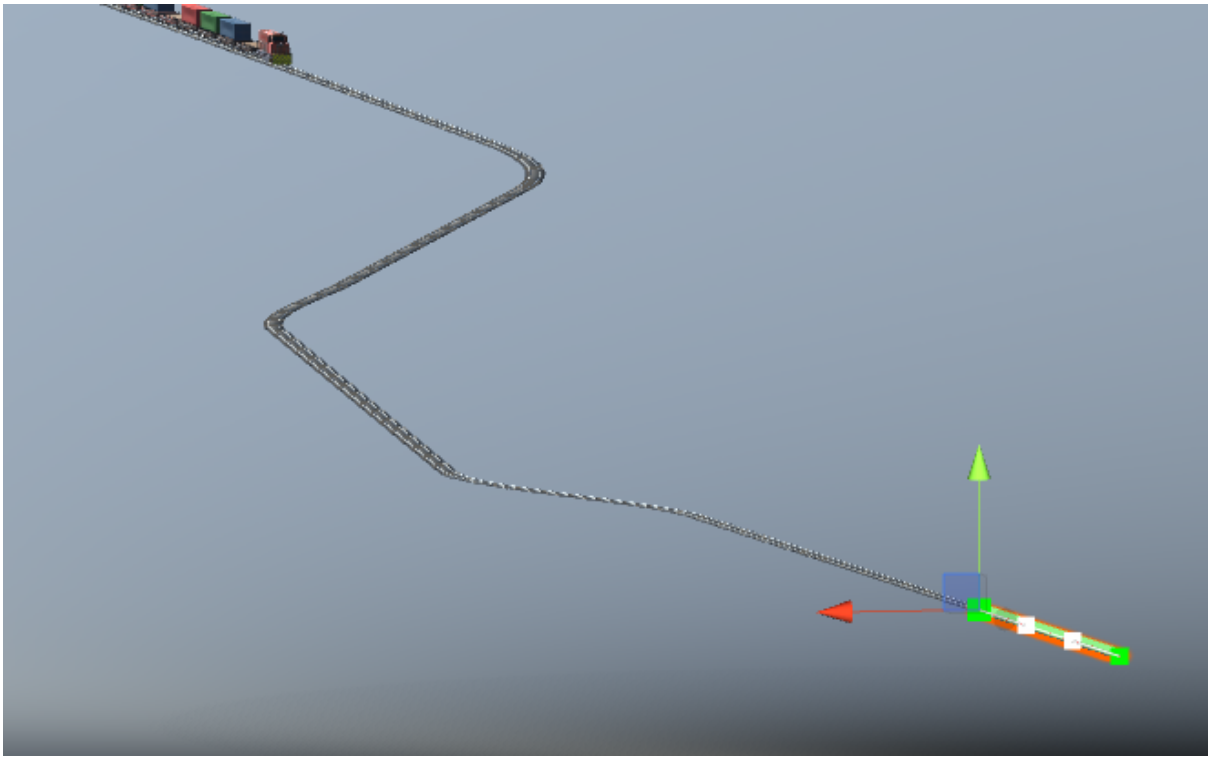
Now, that the warning is gone and the rails mesh matches the last control point. To keep building your super long railroad, just click the "Connect New Renderer" button.



It will create a new Railroad Builder object on your scene aligned at the of the previous one. Use this new Railroad Builder to continue building your railroad.

> **NOTE:** By default, the newly created Railroad Builder will inherit all the settings from the previous one.

Repeat this process as many times as you need. Each Railroad Builder will compose a very long segment of your railroad.

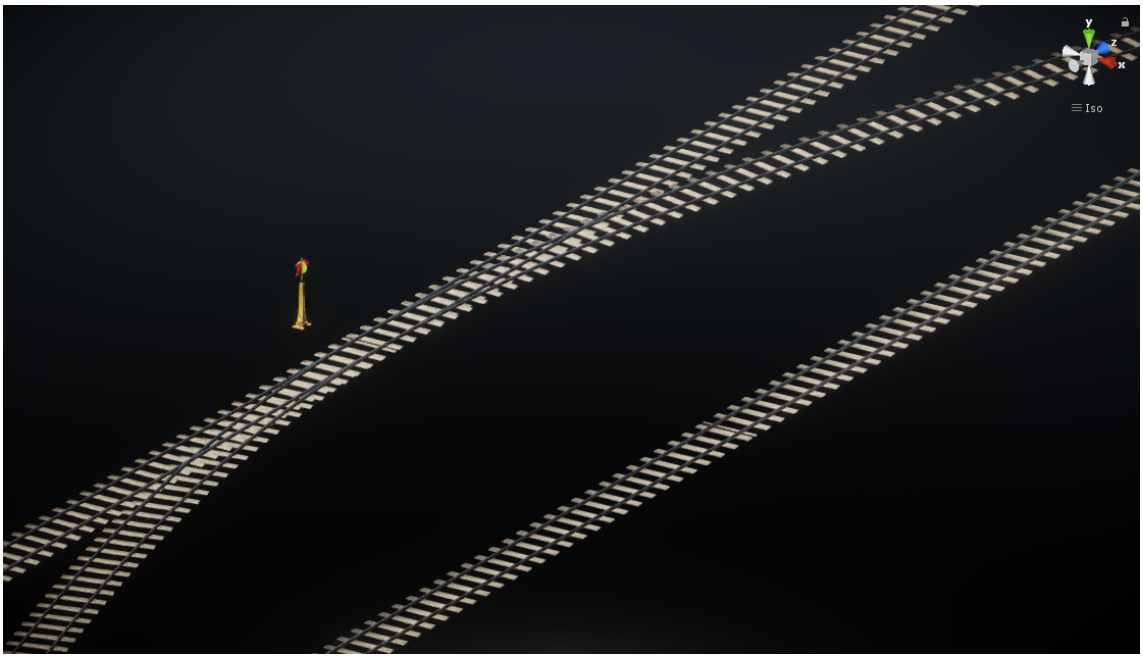


After finishing your railroad, is recommended to export all segments as prefabs using the Mesh Baker and replace all Railroad Builders with baked rails prefabs to improve scene performance. (See the Railroad Prefab Export section for more details)

8. Railroad Switches (Turnouts)

In this section you will learn how to use, signalize and create custom Railroad Switches using the “Railroad Turnout Creator”.

You can also learn this by watching this Video Tutorial instead.



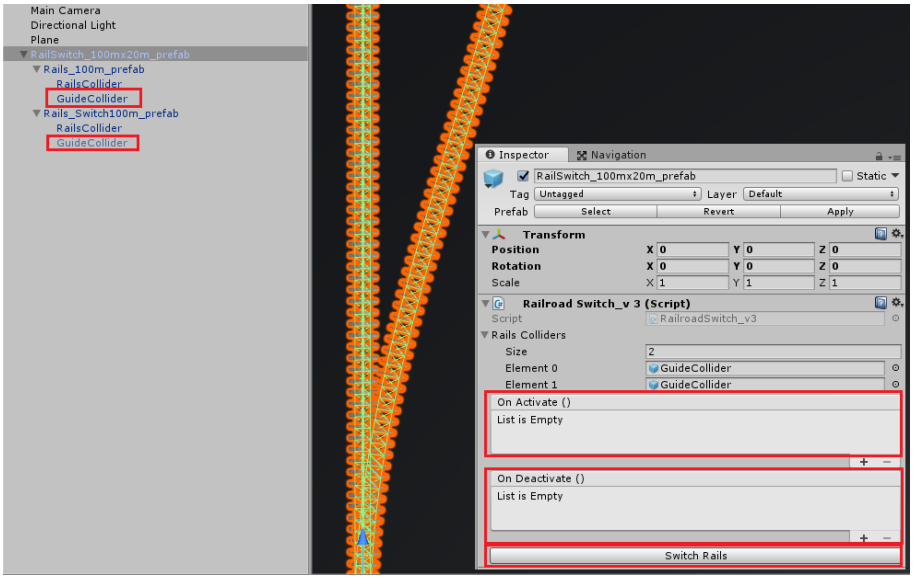
You will find sample turnouts at “Prefabs/Turnouts” folder. There is also, several track switching demo scenes at “Demo” folder.

8.1. How it works

Railroad switches works by enabling/disabling the rails “Guide Colliders” to redirect the trains on the desired direction. This behaviour is controlled by the “RailroadSwitch_v3” script.

You can also change the initial direction of the rails at the Unity Editor by using the “Switch Rails” button on the inspector.

The “OnActivate” and “”OnDeactivate” event stacks can be used to trigger custom events on your game as you wish. They are also used to sinalization (see Railroad Turnout Sinalization section for more details)

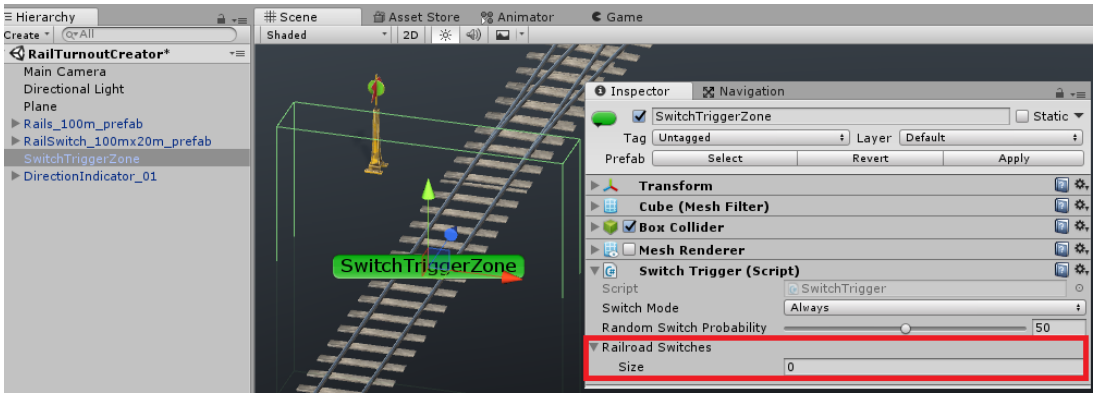


Switches are activated automatically when a train enters a “SwitchTriggerZone”. But, can also be activated manually by calling the “SwitchRails” method of the “RailroadSwitch_v3” script.

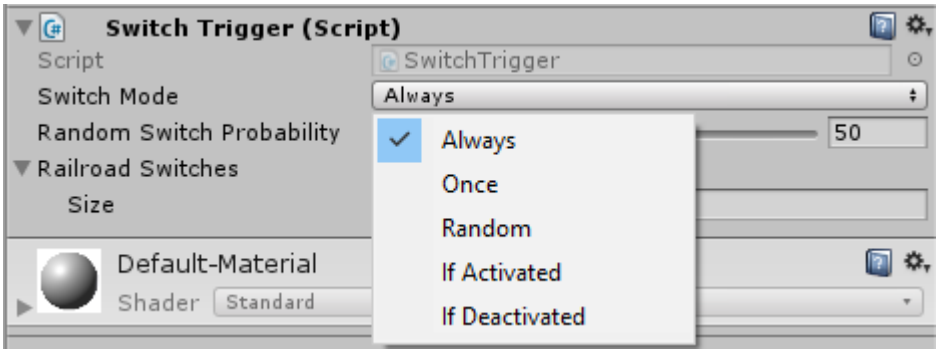
8.2. Railroad Traffic Control

The “SwitchTriggerZone” prefab can be found on “Prefabs/Control Zones” folder. It is controlled by the “Switch Trigger” script and can activate one or more Railroad Switches at the same time.

You just need to drag & drop the railroad switches you want to control to the “Railroad Switches” property of the Trigger in your scene.



Switch Trigger Zones are a powerful way to automate your railroad. It allows to create custom complex railroad traffic control within minutes, just by selecting the desired switching mode.



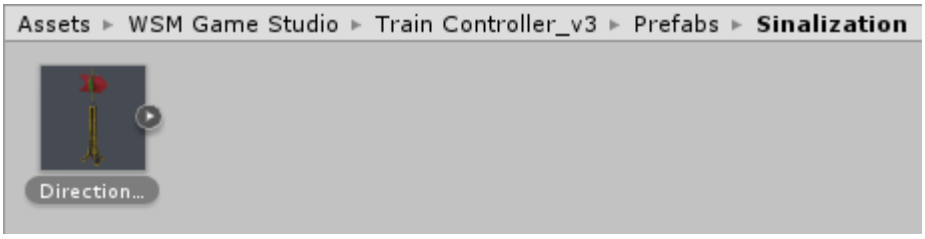
You can choose to switch the direction of the rails every time a train hit the trigger (always), only once, randomly or only if the turnout is activated or deactivated.

Random activation is controlled by the “Random Switch Probability” propertie, so you can fine tune the switch to activate more or less often.

“If Activated” and “If Deactivated” are useful to avoid derailing by creating safety check triggers before each railroad turnout.

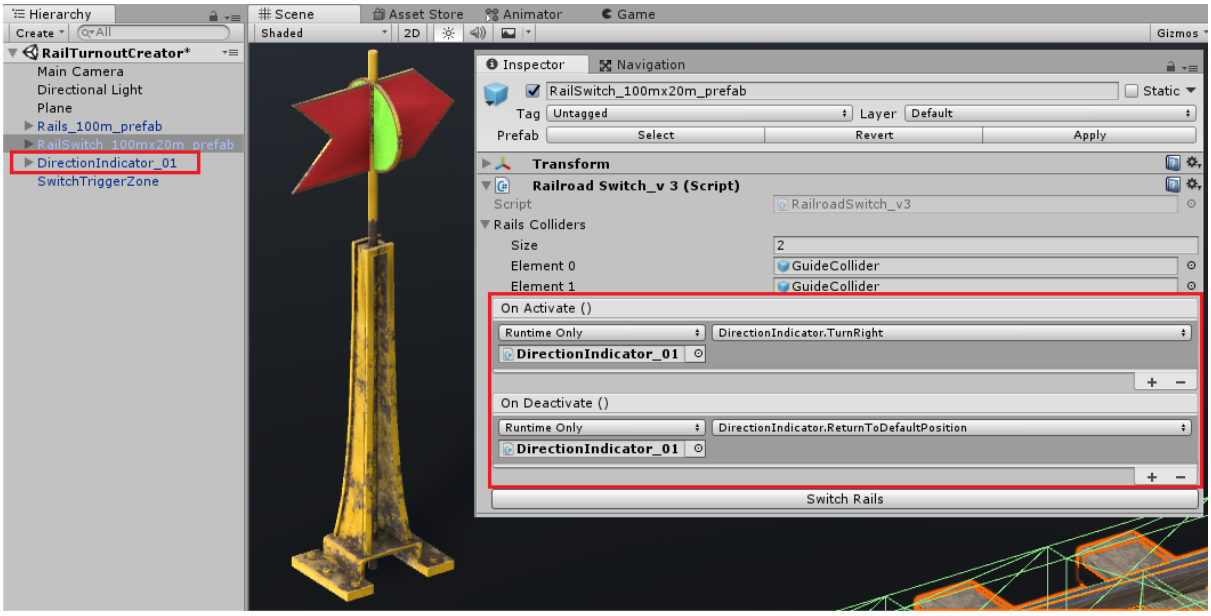
8.3. Railroad Turnout Sinalization

At the “Prefabs/Sinalization” folder you will find the “DirectionIndicator_01” prefab.



It is a ready to use animated direction indicator based on old real world direction indicators. It can point right, left or forward (default position)

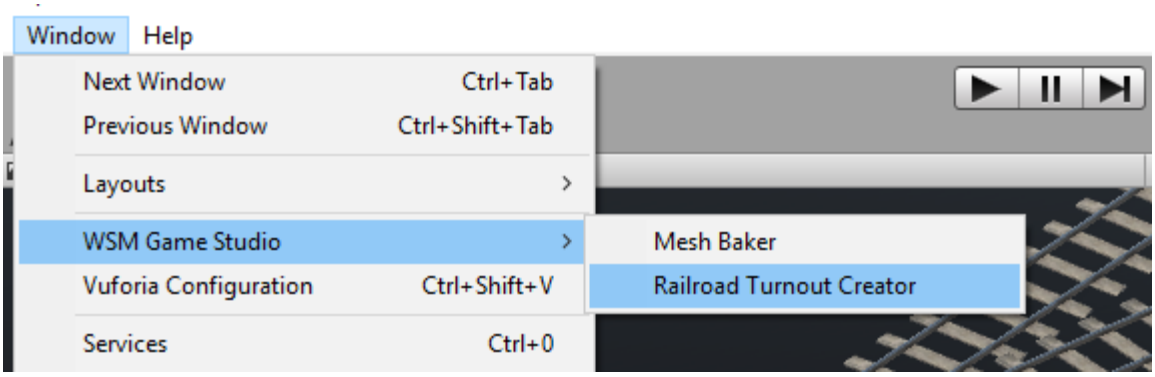
Drag & Drop it to your scene and place it next to the railroad switch you want to sinalize. Select the railroad switch object on your scene and use the “OnActivate” and “OnDeactivate” event stacks to call the “TurnRight”, “TurnLeft” or “ReturnToDefaultPosition” methods like in the sample image below.

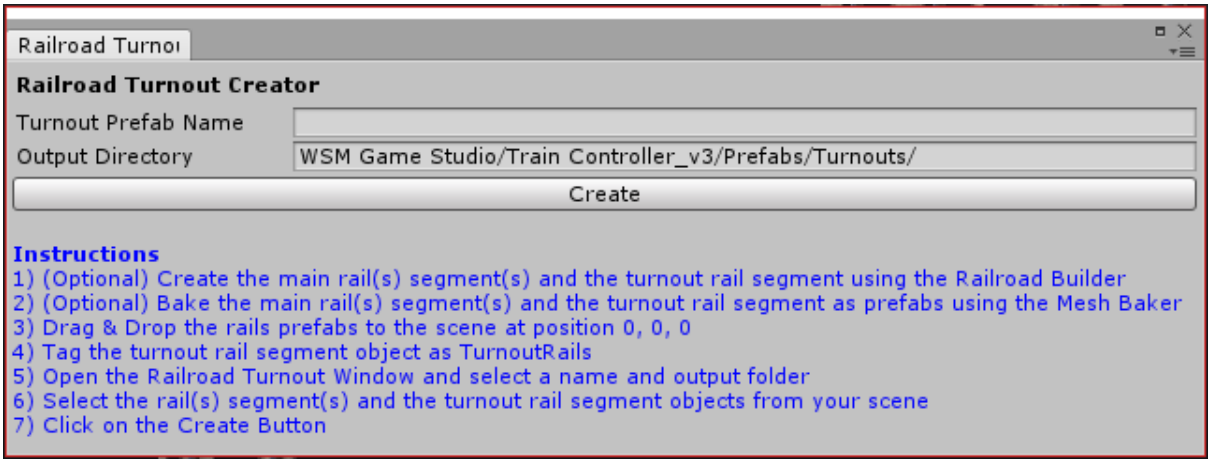


8.4. Custom Railroad Turnout Creator

You can create custom railroad switch prefabs by using the built-in Railroad Turnout Creator.

You can also learn this by watching this Video Tutorial instead.

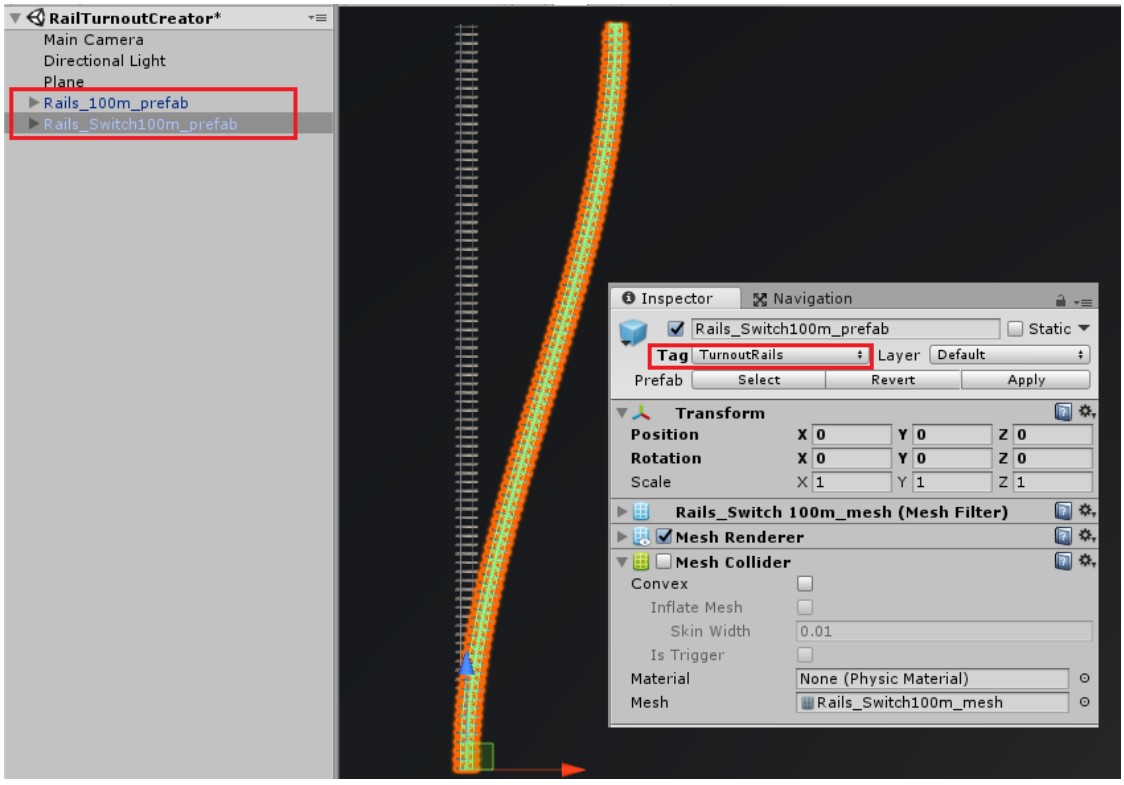




To create a custom turnout, you just the rail segments that will be used to build the turnout. Custom rail segments with any length or curvature can be created by using the Railroad Builder and Mesh Baker (See Building a Railroad section for more details)

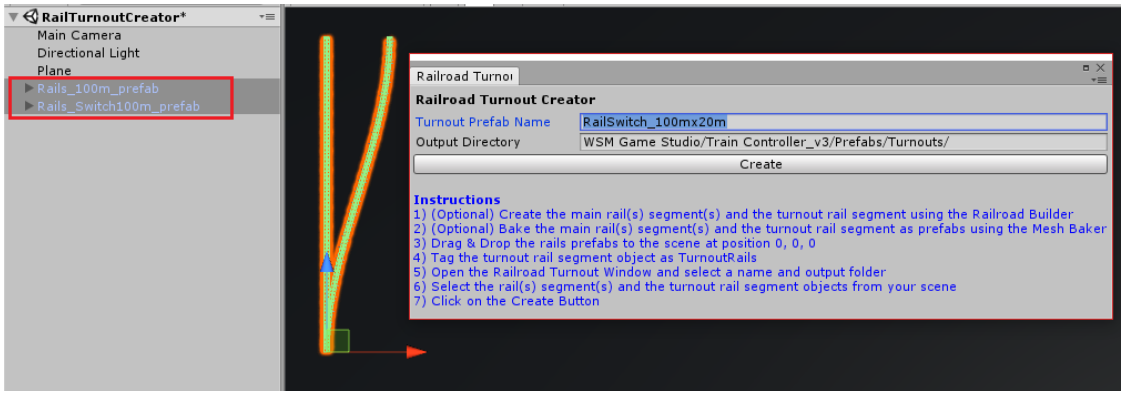
Drag & Drop the rail segments prefabs to your scene and set their positions to 0,0,0.

Then, select one of the segments (preferable the curved one) and set its Tag to “TurnoutRails”. Only one segment must be tagged as “TurnoutRails”. The tagged rail will be disabled by default.

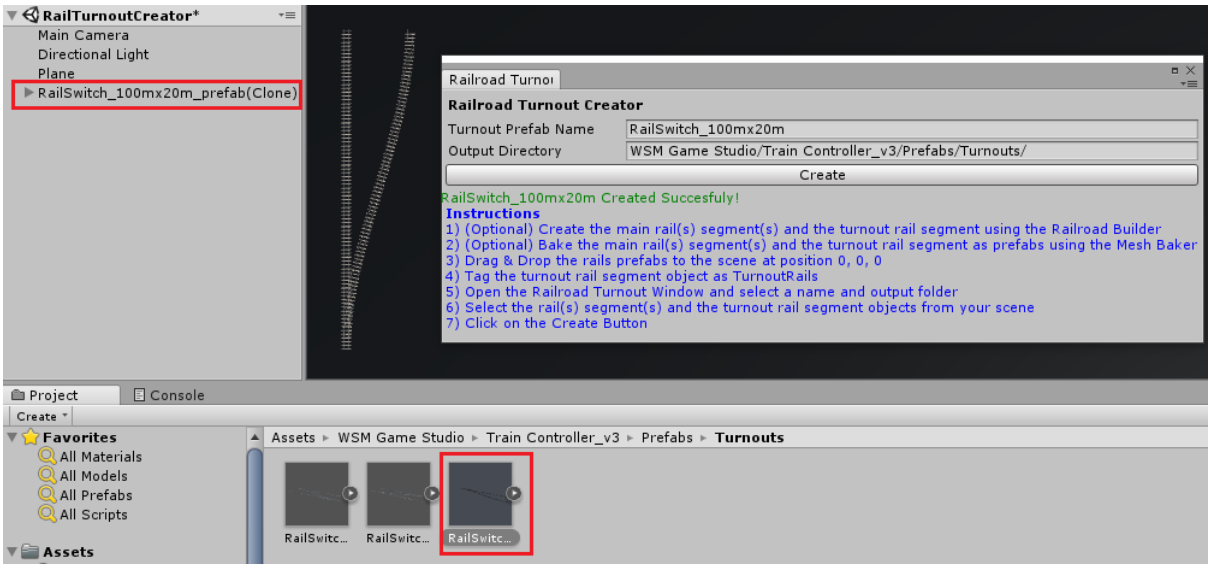


Open the Railroad Turnout Creator Window (Window > WSM Game Studio > Railroad Turnout Creator) and choose a prefab name and output directory.

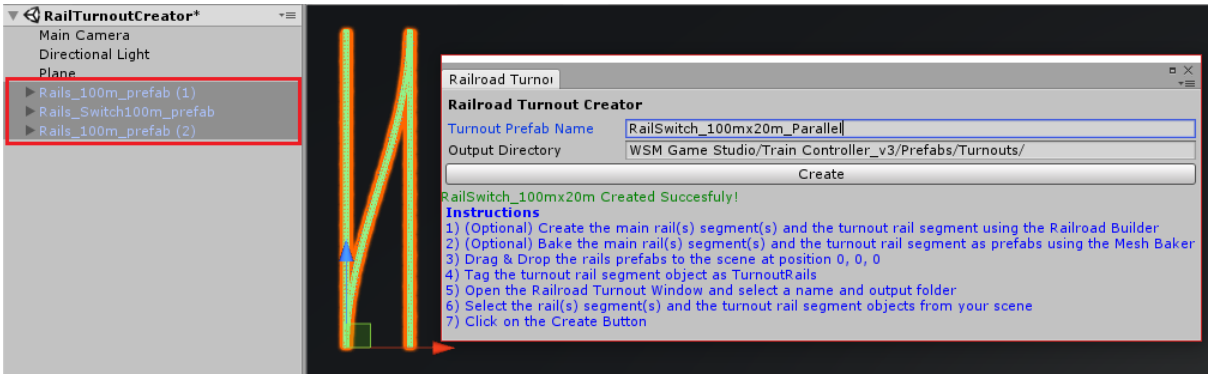
Select the rail segments on your scene and click on “Create”.



If done correctly, the prefab will be saved on the selected output folder and a clone will be instantiated on the scene.



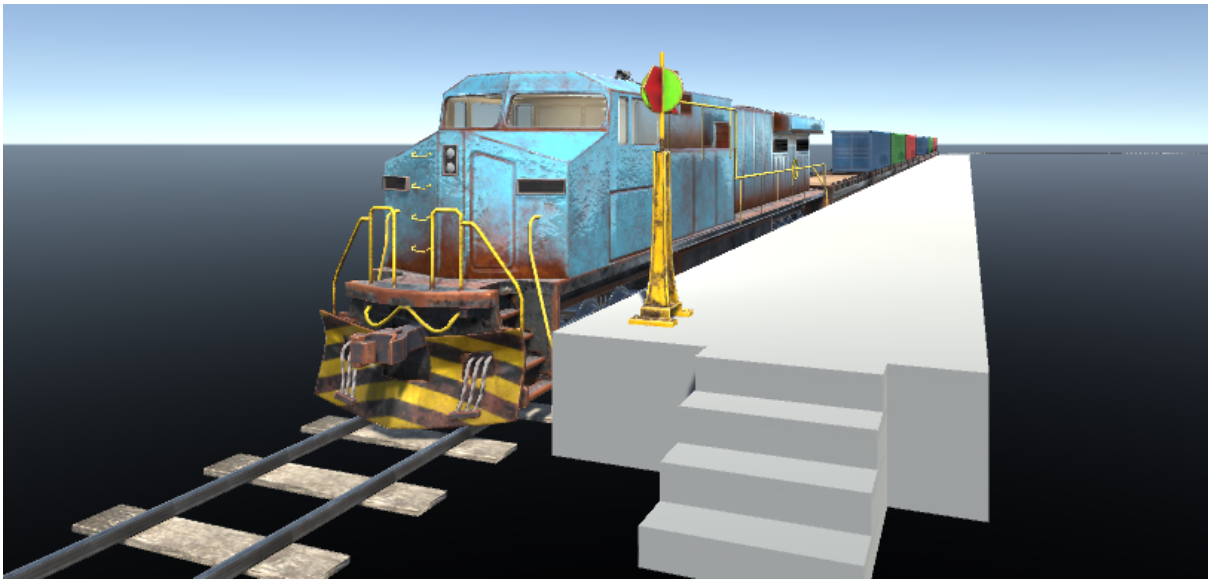
To create parallel rails connected by a turnout, you just need to add another rail segment, adjust its position and select all three rail segments before clicking on create (In this case, only the middle segment should be tagged as “TurnoutRails”).



9. Train Stations

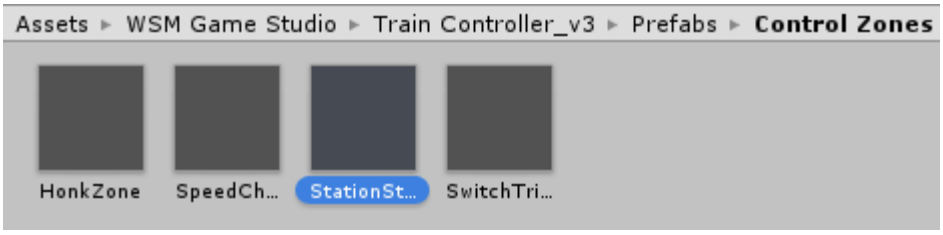
In this section you will learn how to simulate train stations

You can also learn this by watching this Video Tutorial instead.

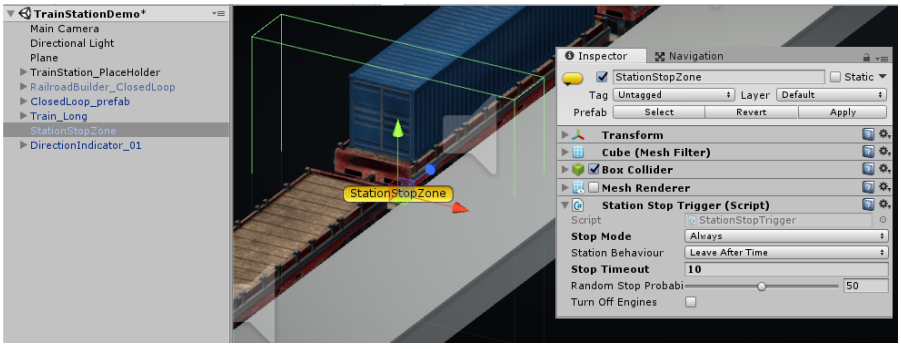


9.1. Stopping at the Station

In the “Prefabs/Control Zones” folder you will find the “StationStopZone” prefab.



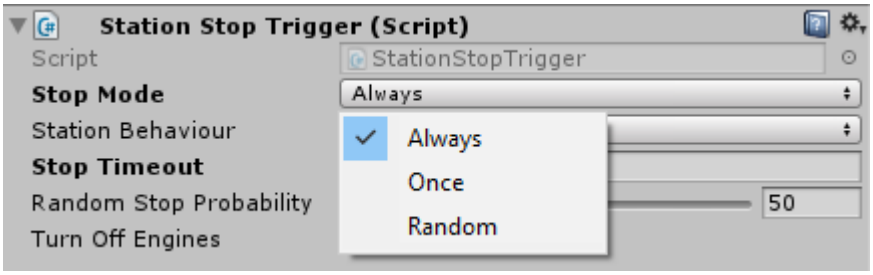
It can be used to simulate train station stops. Once a train hit a “StationStopZone” trigger, it will start braking until it stops.



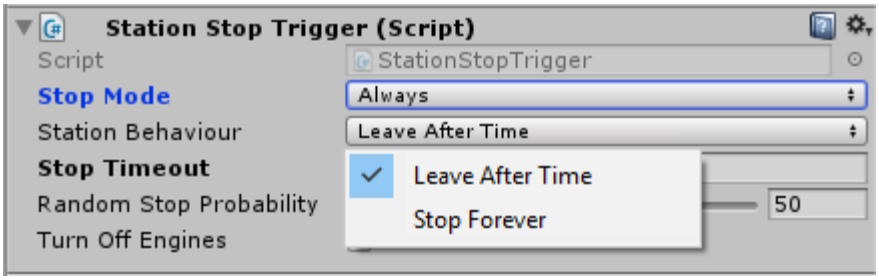
You can configure when the train should stop and how it behaves at the station after stopping.

You can choose to always stop, stops only once or randomly stops at the station.

Random activation is controlled by the “Random Stop Probability” propertie, so you can fine tune if the train will stop more or less often.



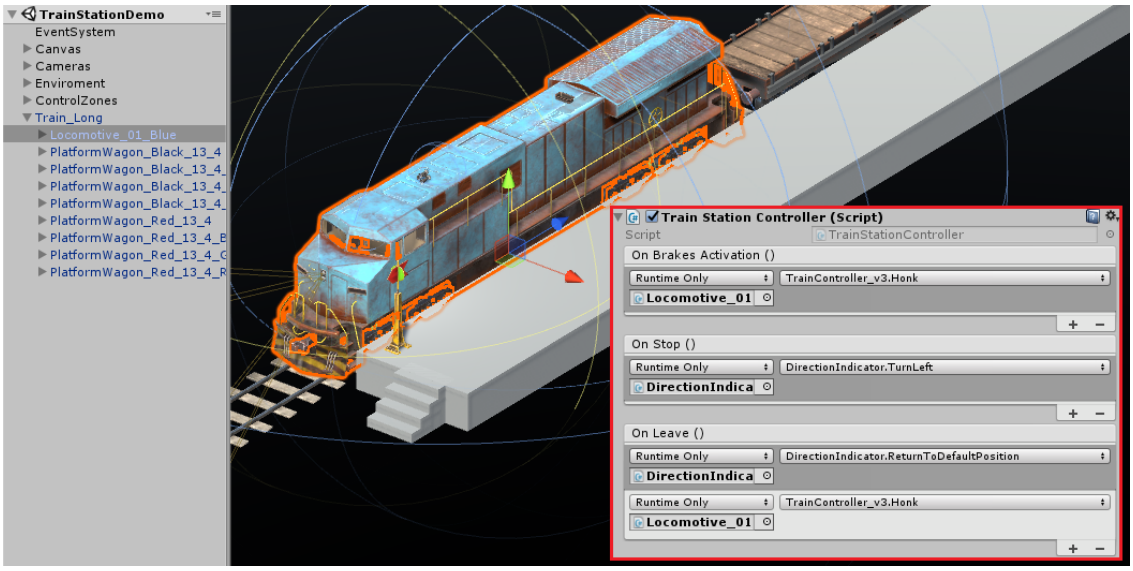
You can also choose to leave the station after some time or stay forever. The “Stop Timeout” property defines how many seconds the train will remain on station before leaving.



9.2. Station Custom Events

You can call a custom event stack whenever the train starts breaking, stops moving or leaves the train station.

Custom events should be set at the **Train Station Controller** script of each train. This allows each train to trigger different events at the scene if needed.



9.3. Train Doors

This package includes train doors open and close Mechanim animations.

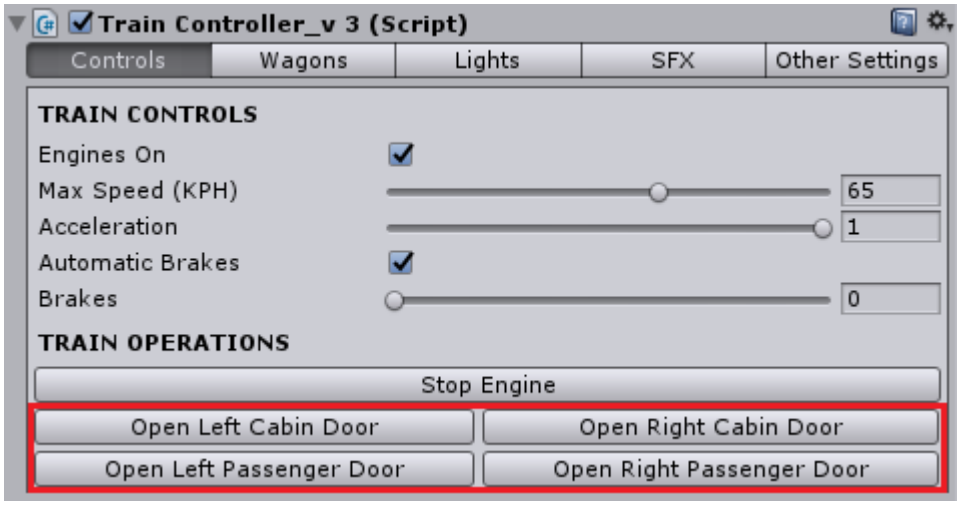
Since animations were made with Mechanim, this means you will be able to customize the animations in the Unity Editor if you wish.

By default, when the subway reaches a station it will automatically open the passenger doors (See the Train Stations section for more details).

But, if you wish to open/close the doors at any other time, you can call the open/close methods directly from a button click event or from a custom script for example. All you need to do, is to call the methods from the **TrainDoorsController** script attached to your locomotive subway cart.

- **TrainDoorsController** script methods
  - Open Methods
    - OpenCabinDoorLeft()
    - OpenCabinDoorRight()
    - OpenPassengerDoors()
  - Close Methods
    - CloseCabinDoorLeft()
    - CloseCabinDoorRight()
    - ClosePassengerDoors()

You can also call this methods in the Unity Editor by using the buttons available on the **Train Operations** section of the **Train Controller** script. This is useful to test your custom train doors.



NOTE: Door related operations are only enabled on **Play Mode**.

10. License

By purchasing this asset you are allowed to use it for unlimited games and/or 3D projects (like animations, simulation softwares, etc). Both personal and commercial use.

You are **NOT** allowed to resell or distribute the assets components individually or as part of another asset package (including, models, scripts, etc).

11. Contact Info & Support

If you have any questions, need support or have some business inquiries, feel free to get it touch.

The best way to reach us is by email on **wsmgamestudio@gmail.com**.

Asset Store
Discord
Sketchfab
Instagram
Facebook
Twitter
Youtube Channel