Configure and script couplings

Table Of Contents

1 Configuration

You configure couplings in the object properties of a vehicle.

Down below there is the section "coupling front" and "coupling back". These have all necessary parameters:

- Y coordinate: At this point the point gets defined where both couplings attach each other.
- Mount: Coordinates of the point where the coupling is mounted at the rail car body
- Parent animation: Select the animation, where the clutch bearings are attached to.

2 Animation

Couplings can be animated. So you can rotate a Scharfenberg coupler in the right direction to match both couplings. The Content Tool provide the animation type "Coupling".

The process of configuration is the same as the configuration of other animations:

- Create a new animation (one per coupling) of the type "coupling"
- Edit the properties of the couplings: The index of the coupling should be 0 for front couplings, 1 for back couplings
- Mit "Meshs hinzufügen/entfernen" die zu animierenden Meshs hinzufügen.
- Attach the mesh you want to be animated with the "Add/delete meshes"

Of course you can add animations as "childs" of the coupling animation.

3 Script

3.1 Coupling state

The coupling state of the front and back couplings can be checked with the variables `coupled_0` and `coupled_1`. There is no need to declare these variables within the script.

3.2 Couple and decouple via script

If vehicles couple when colliding or the decoupling of two vehicles are controled by the variables: `couplingState_0` und `couplingState_1`. These variables can have the 3 following states:

- 0: If the state is 0, the couplings get decoupled and will remain decoupled even if the vehicles collide. So the other vehicle will be pushed away, but you can't pull the other vehicle anymore. This is the function of a screw coupling with bumpers and hooks, which is standard on european trains. When the vehicles collide and the variable is set on the value of 2 the vehicles couple.
- 1: This is the standard state and describes the decoupled state of an automatic center buffer coupler like the Scharfenberg coupler used by trams, metros, suburban trains and so on. This coupling is defined to couple when colliding with an other coupling.
- 2: The vehicles are coupled. This value will be set automatically when both couplings had the value 1 before colliding. If this condition is not set, the automatic coupling won't work.

TL;DR: To decouple two vehicles with an automatic coupling simply set both variables on "1". If the couplings collide afterwards again, the vehicles will couple again.

3.3 Send data between to vehicles

3.3.1 Send

There are four functions to send data at coupled vehicles:

- SendToECouplerSingle(self: integer; id: string; value: single; couplingindex: integer),
- SendToECouplerString(self: integer; id: string; value: string; couplingindex: integer),
- SendToECouplerInteger(self: integer; id: string; value: integer; couplingindex: integer) and
- SendToECouplerBoolean(self: integer; id: string; value: boolean; couplingindex: integer).

The first parameter has to be the (automatically declared by LOTUS) public variable "Self". The couplingindex is zero when this is a front coupling and 1 if it's a back coupling.

3.3.2 Recieve

On the recieving side you have to declare variables the broadcasting vehicle can use. These variables have the following name convention:

- [id]_front and [id]_back recieve the broadcasted value (could be either single, string, integer or boolean) (recieved either on the front coupling or the back coupling)
- [id]_front_sent and [id]_back_sent will be set to true after the transmitting procress and after the script is done they will be set to false.

3.4 Alternative communication between vehicles

The module system brings the principle of broadcasts between vehicles and modules. However, these broadcasts can also be enabled so that they can also run over couplings, which provides another way in which coupled vehicles can communicate with each other.

More details can be found in this article: https://www.lotus-simulator.de…l-objekte/#3.2-Broadcasts