Manual to Plug-in 1 for EEP 16.1

Dear customers, first of all we would like to heart-fully thank you for purchasing EEP 16.1 Plug-in 1.

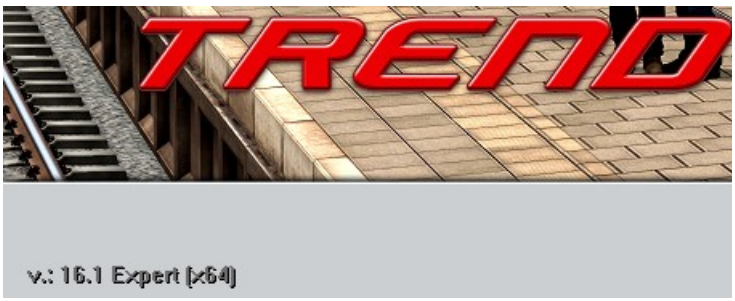
This plug-in includes both improvements and new functions. For this reason, we invite you to read the following information carefully and in particular the installation instructions.

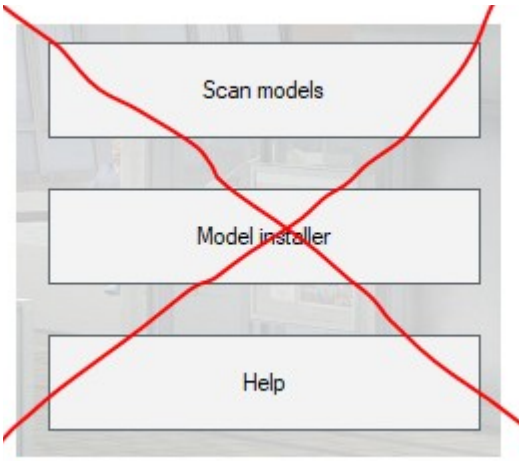
Glossary:

Manual to Plug-in 1 for EEP 16.1

Installation instructions

Please make sure that update #1 of EEP 16 is installed. This is the prerequisite for this plug-in. You can check this in the lower left part of the loading window. If your system is running on a 64-bit architecture, the mention (x64) will be added to the running version.



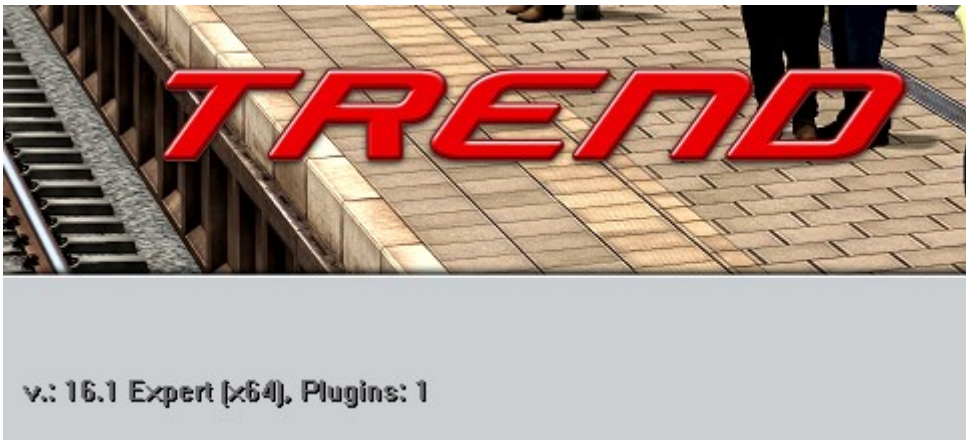


The Plug-in needs its own installer to upgrade, since it brings new features besides the new models. For this reason it is not possible to open this file directly via the "Model-Installer of EEP16.1

If necessary, please leave your EEP 16.1

Please start installing the Plug-in by double clicking on the file V16TSP10041. A message informs you that this program will operate changes to your computer. Please allow this by clicking Yes to proceed with the installation. The installation window then appears and prompts you to accept the user license and once these steps are completed, the program then extends the functionality of your EEP 16.1

If the installation was successful, then you will see in the lower left part of the launch window the EEP 16.1 version followed by the mention Plugins 1



Manual to Plug-in 1 for EEP 16.1

New models contained in Plug-in 1 for EEP 16.1:

New equipment for the bridge crane:

- Hook
- Magnet

New goods adapted to the new loading possibilities:

- Steel beam 40 x 28 x 500 cm.
- Pallet box – 1 hook point
- Log bundle fir x3
- Log bundle fir x3 (short)
- Log bundle spruce x3
- Log bundle spruce x3 (short)
- Steel pipe 60 x 500 cm.

Structure:

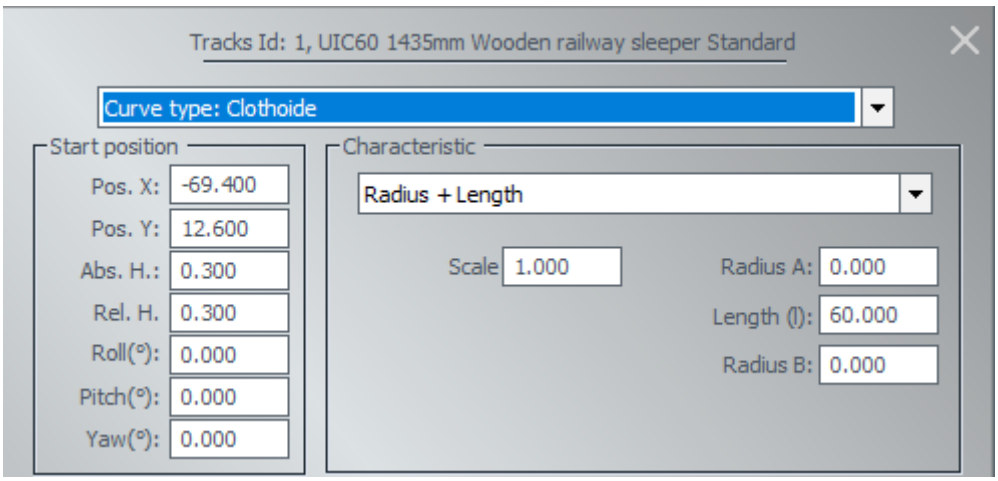
- Railway station Baden-Baden

New features allowed by Plug-in 1 of EEP 16.1:

The new curve called Clothoid

The clothoid is a flat curve with a curvature that gradually increases. It is particularly useful when passing between a bend and a straight line or between two bends with different curves.

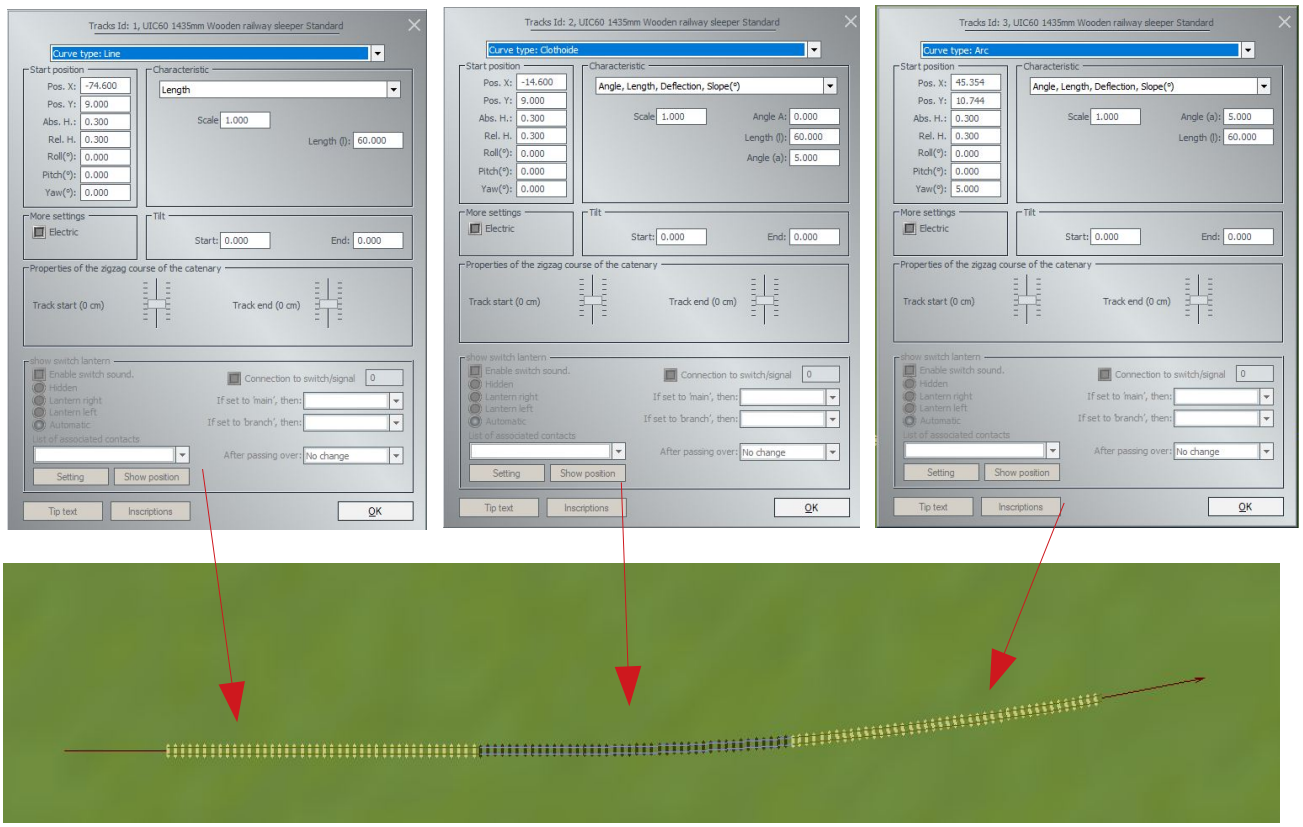
The clothoid can be defined either by using
radius + length
angle + length



A angle or A radius reveals the curvature at the beginning of the track, while the B angle or B radius reveals the curvature at the end of the track.

Application example:

I would like to pass smoothly from a straight section to a 5° bend.



To my first line rail, I add a second clothoid rail with a start angle of 0° and an end angle of 5°.

I therefore enter the angle A = 0 and angle B = 5.

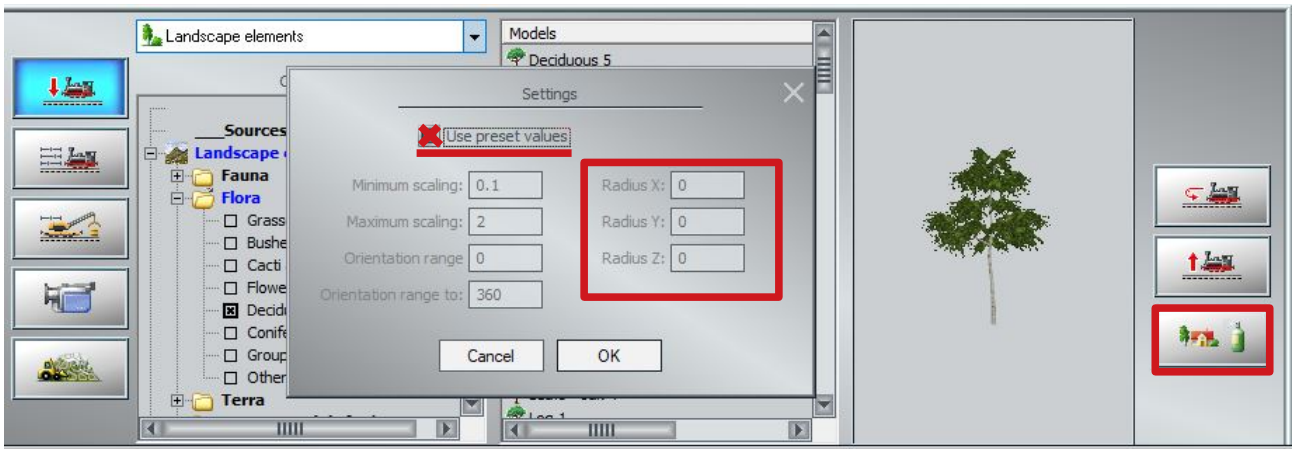
Then comes my 3rd rail, it's an arc with a constant curvature of 5°.

The menu for implementing a model according to predefined rules now allows you to add a random offset for the radius.

With Plug-In 1 for EEP 16.1, we extended once again the random variations that can be applied to buildings and landscape elements when they are installed. (see 4.5.2 of the EEP16 manual)

You can now implement these models in a random orientation with respect to the X, Y and Z axes.

To use this function, open the dialog window and check the option "activate predefined values". That way you can enter values for scale variations, orientation variations but now also insertion limits with a radius relative to the X, Y and Z axes.

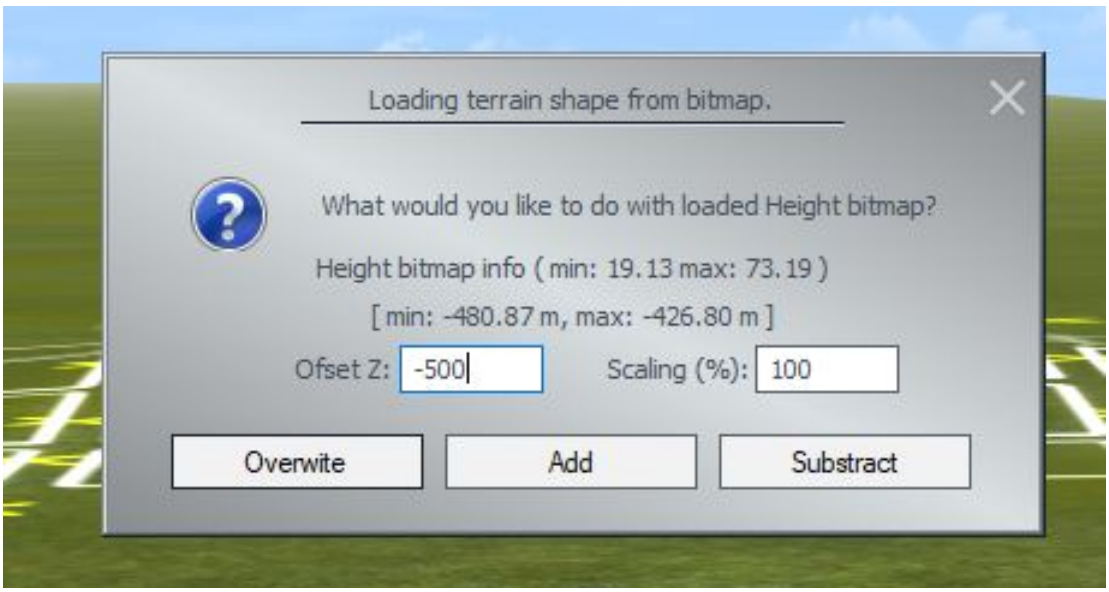


Scaling of Bitmaps relief files in 2D and 3D display - extended to 16Bit Bitmaps

The landscaping possibilities using relief maps have been extended with Plug-in 1 for EEP 16.1 (see 4.3.4 of the EEP16 manual).

Until now, it was not possible to adapt the preregistered relief maps in height and they had to be superimposed on each other until the desired height was reached. Now it is possible to increase their scale or define an initial offset on the Z axis.

This avoids having to read several relief bitmaps files. In addition, 16-bit bitmaps are now also supported, enabling you to create an even more detailed landscape.



Shifting a Bitmaps relief file

The height offset of Bitmaps format relief files allows, for example, to modify the minimum and maximum height of the relief in order to adapt it to the project's relief's height.

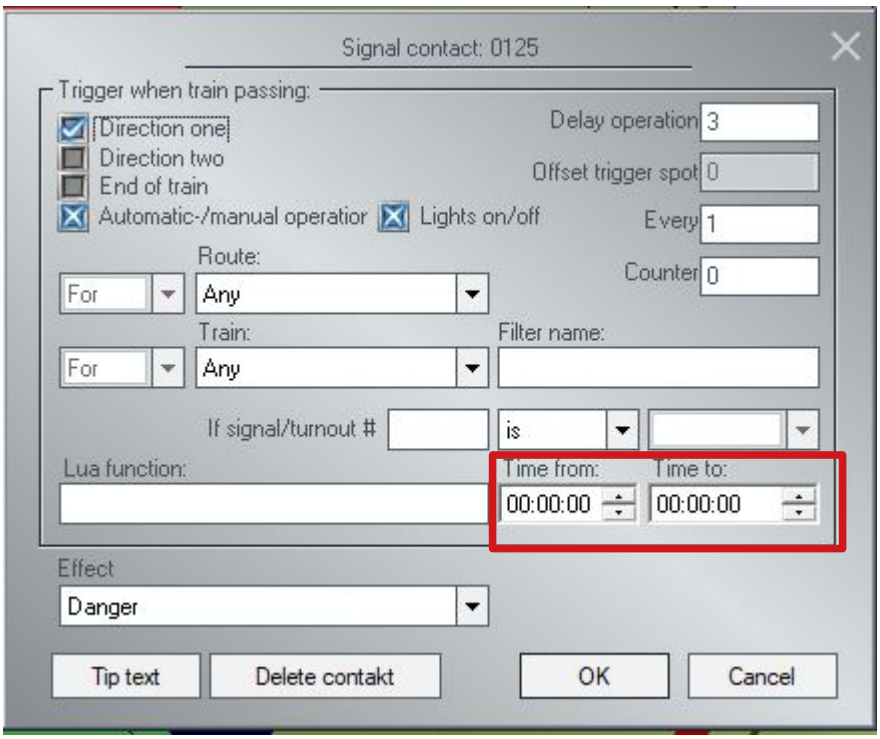
By entering a Z-shift value of -500 meters, the heights of the Bitmaps relief file, which for example were 500m above sea level for the lowest value and 1500m above sea level for the highest value, will then reduce to 0m above sea level and +1000m for the highest peak, once this file is inserted into the project.

The execution of commands defined by contacts can now be enabled or disabled over time.

As a response to the expectations of many EEP users, the functionalities offered by contacts have been extended. Thanks to Plug-in 1 for EEP 16.1 it is now possible to activate a signal according to a predefined schedule.

To do this, please fill in the fields relating to the time slot during which the signal must perform the requested function. Be careful, we are talking here about EEP time.

This way you can activate or deactivate a switch at nightfall, which can be very useful.

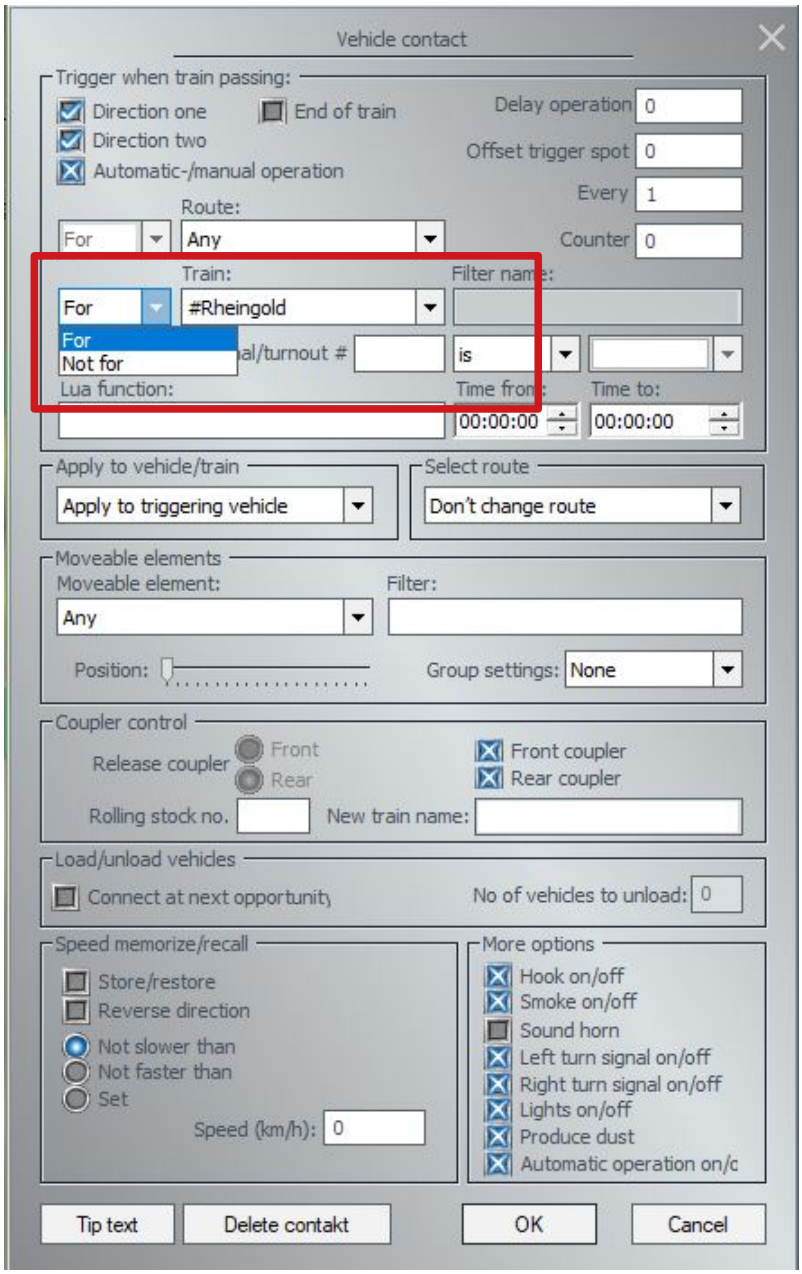


This ability is now available for all types of contacts.

Contacts now act selectively on the objects to be controlled using the "is not" filter which is available in the settings.

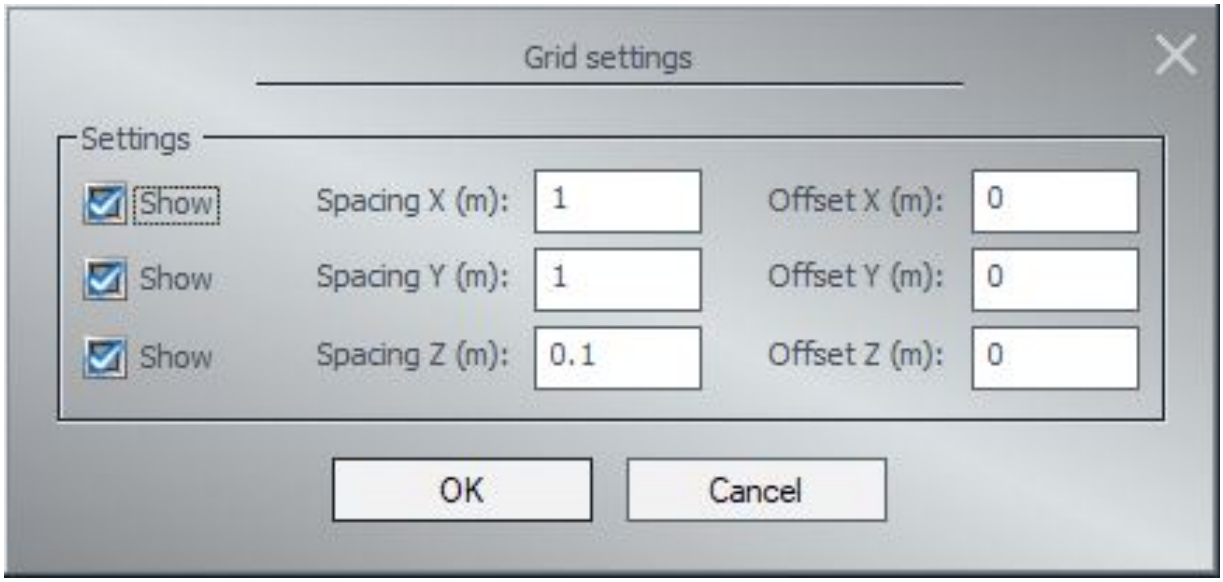
It is now possible to adjust the contacts more precisely by specifying which rolling stock is likely to be able to activate it or not.

Simply enter the rolling stock's name in the red-marked frame and choose whether it has or not to activate the contact.



Shifting the grid and new grid added in 3D display mode

We did also improve the lower menu bar.
Until now it has only been possible to display a grid while being in the 2D planing window (see 2.2.3 of the EEP 16 manual) by pressing the icon above number 3

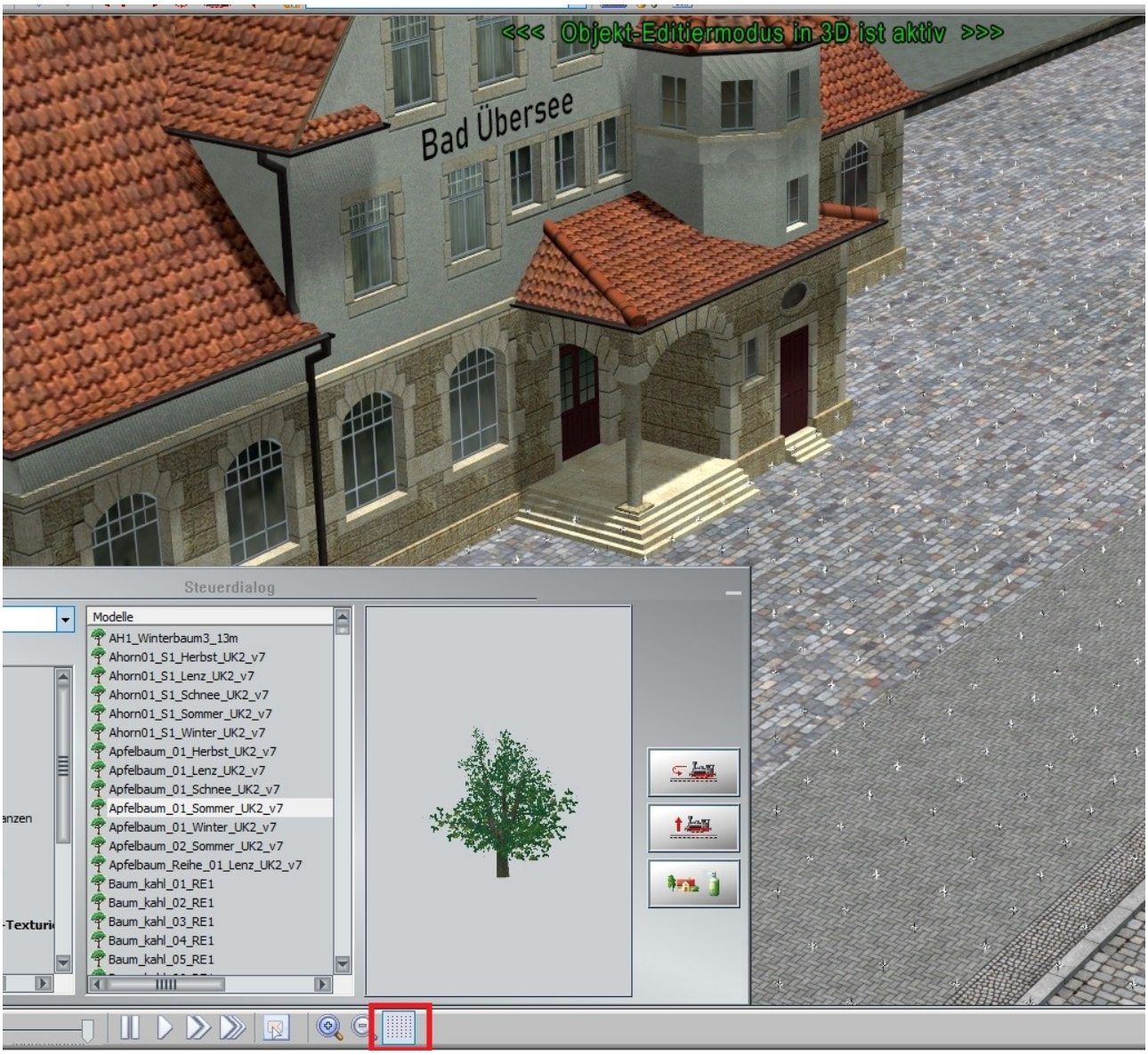


Plug-in 1 for EEP 16.1 allows this grid to be shifted in the plane window along the X, Y and Z axes.

It is now also possible to use this grid, whose properties have been defined in the 2D planing window, by being in the 3D display.

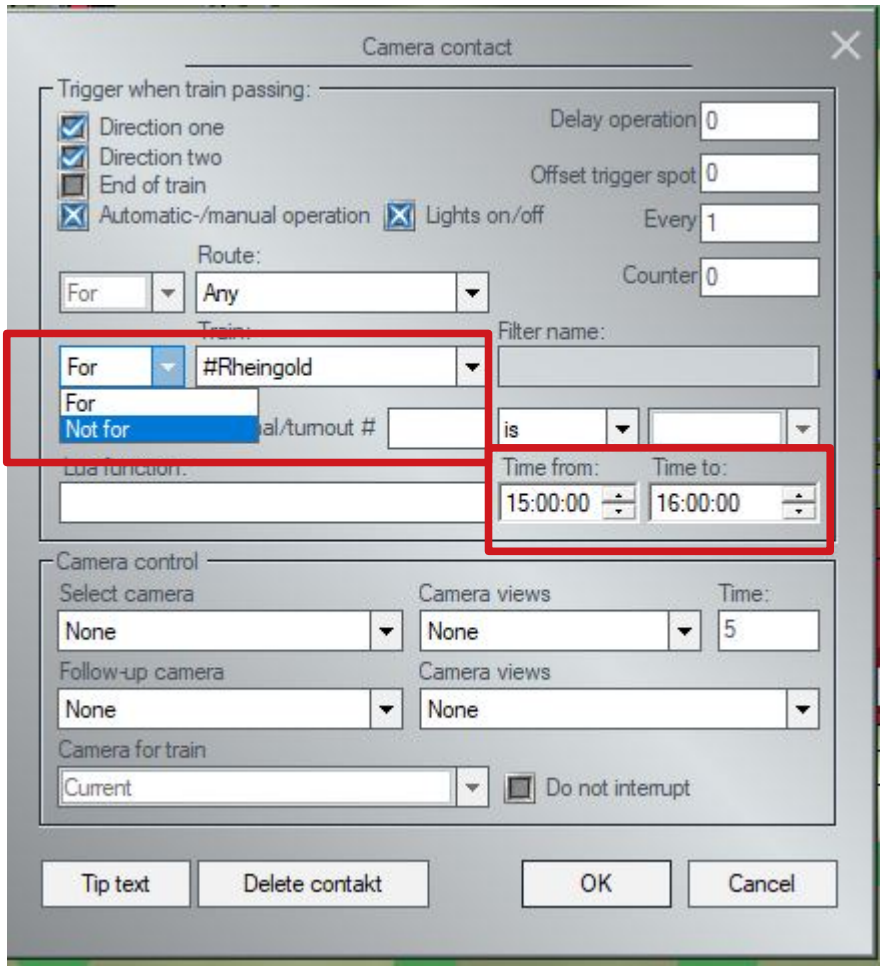
If you want to change the grid in terms of its offset, first go to the 2D planing window and change the X, Y and Z values. Otherwise, go directly to the 3D display.

In the 3D display and making sure you are in the object editor, click on the grid icon. As soon as an object is selected by the mouse, the grid appears.



Contacts for vehicle-specific cameras.

As with conventional contacts, a new feature has been implemented to allow you to extend the camera settings. (see 3.2.5 of the EEP 16 manual) It is now possible to specify whether or not a particular camera should be activated for a given rolling stock.



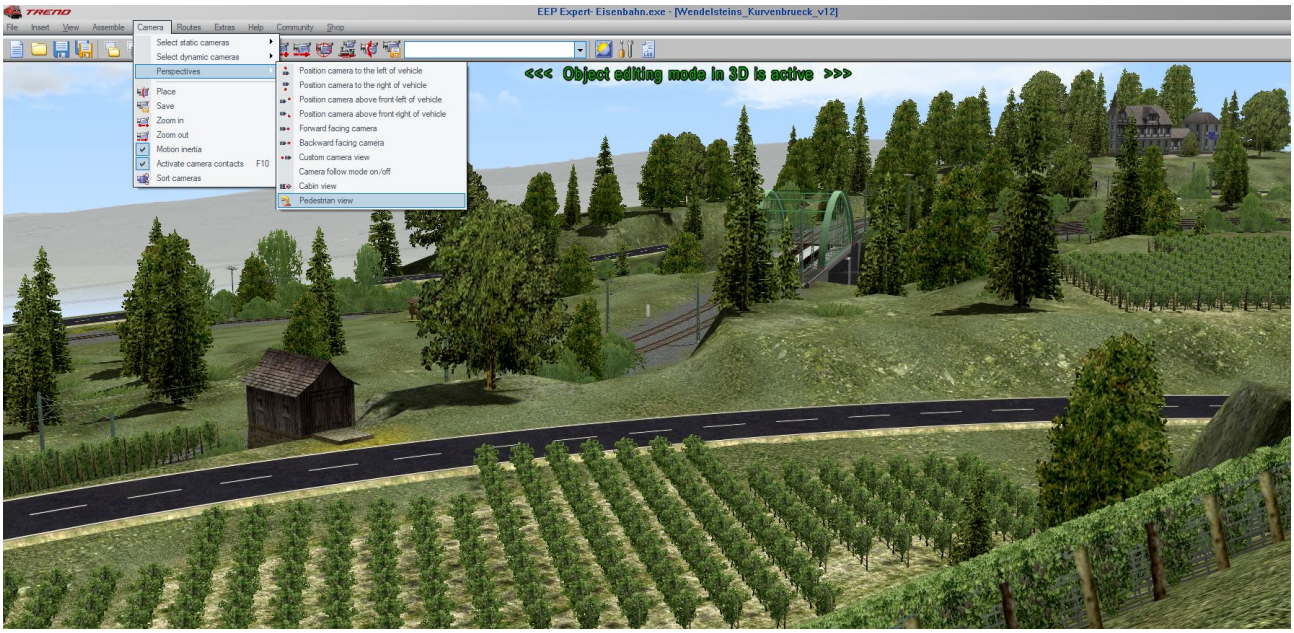
Zoom of the camera

While it was possible to move a camera forward using the space bar, it is now possible with Plug-in 1 for EEP 16.1 to zoom in on an object using the ` key on the left of the 1 key

Manual to Plug-in 1 for EEP 16.1

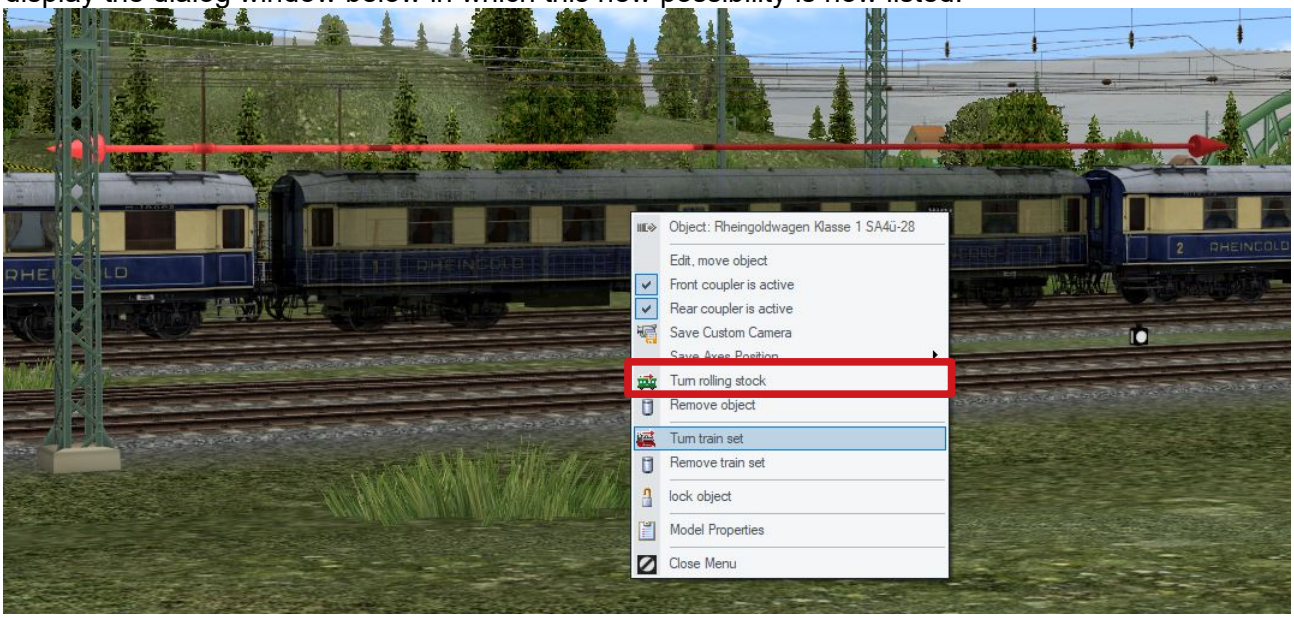
The camera now tilts automatically while being in pedestrian view mode

Among the improvements made by Plug-in 1 for EEP 16.1, we also find the automatic adaptation of the camera angle in the pedestrian view mode. The camera now takes into account the topography and provides a view that is similar to that of a pedestrian walking on the site in question.



New function allowing to return an element in a train set while being in 3D display.

As part of the evolution of EEP we have made it possible to turn over a single element in a train set. To do this, while you are in the control window, right-click on the element to be rotated to display the dialog window below in which this new possibility is now listed.



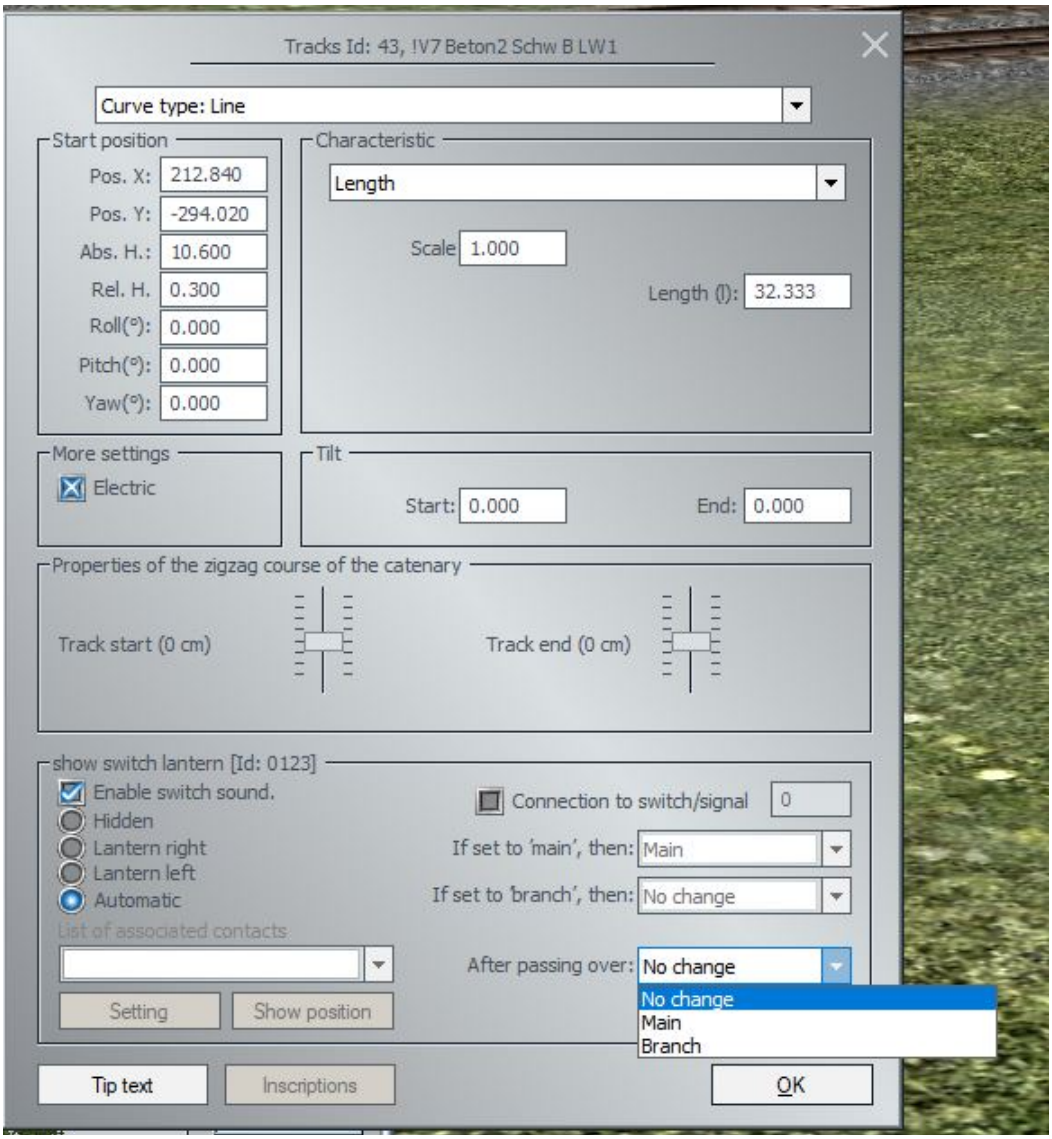
A simple left click on this function and you are done. The selected train element will now face the opposite direction.

New function enabling the switch to return to its initial position automatically after the train has passed.

When using this function, it is no longer necessary to place a second contact to make the switch return to its original position. This function is activated as soon as the last rail car has left the rail with the relevant switch point.

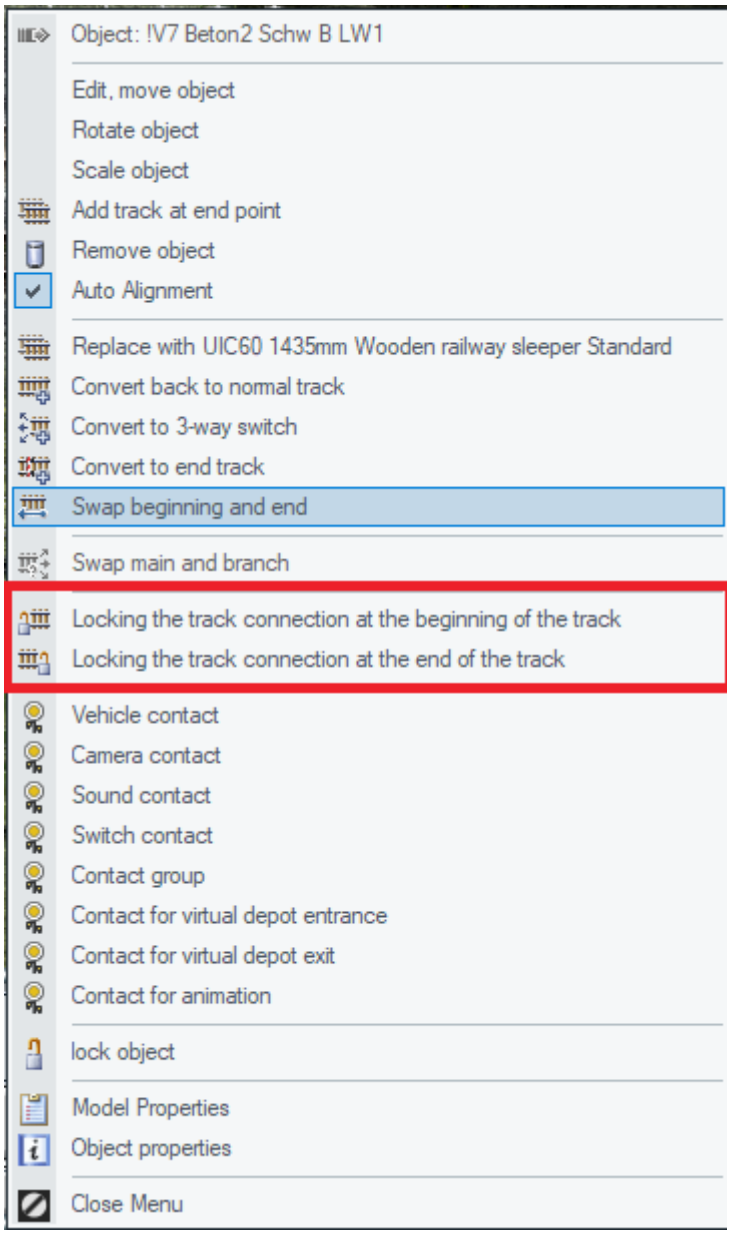
This new feature, implemented by the programmers in this first Plug-in, takes in consideration another wish from EEP users.

Please note that if you try to open this menu using pre-existing track combinations, it will only be possible in the 2D planning window.



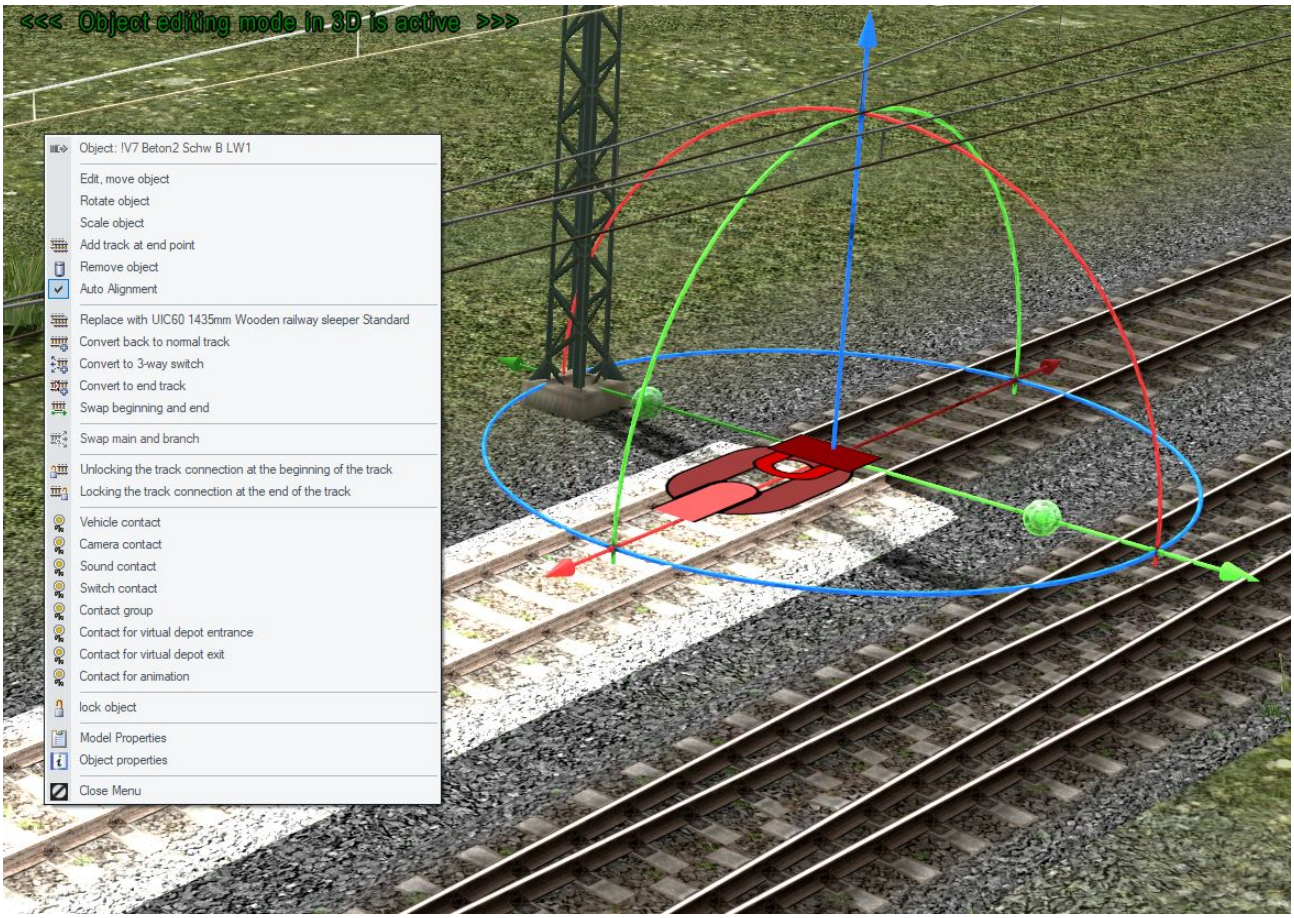
Link locking option at the beginning and the end of tracks.

With Plug-in 1 for EEP 16.1 it is now possible to prevent accidental linking of tracks between them using this new feature available in both the 2D and 3D editors.



To use this feature in both 2D and 3D display mode, right-click on the relevant rail and in its context menu left click to lock either the beginning or the end. The locks at the ends of the rails are indicated in red while being in 3D display mode.

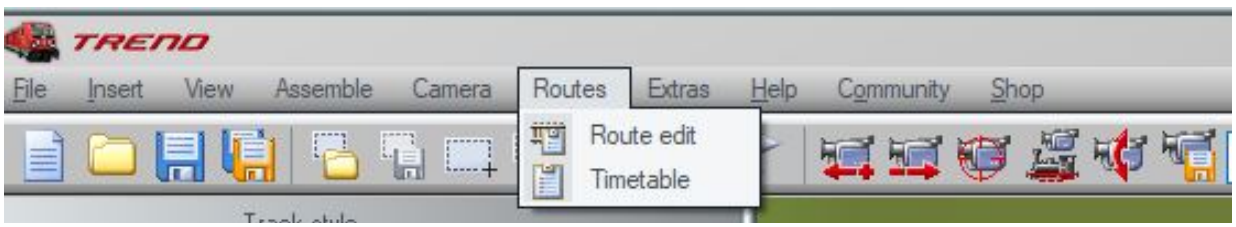
To unlock, please do the same by selecting the relevant rail with a right click.



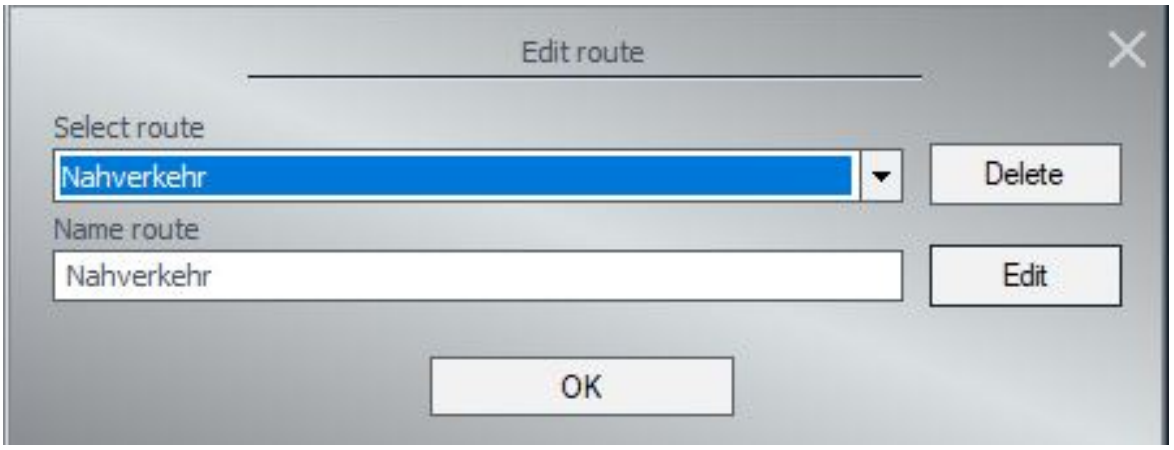
The new filter in the menu routes allows you to show only the contacts on a given road.

Thanks to Plug-in 1 for EEP 16.1, you can now hide all non related contacts and only show those who are relevant to a specific route. (see 6.3.1 of EEP 16 - Road definition and installation).

This filter can be activated by going from the 2D or 3D display to the "Routes" menu and then "Route edit", then simply enter the desired route and confirm with OK.



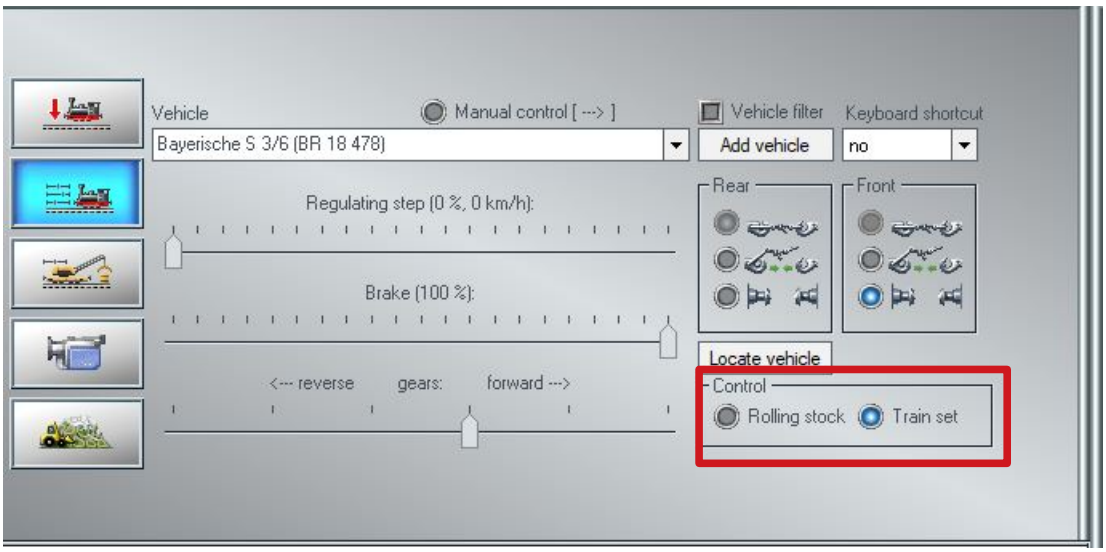
To cancel this filter you only need to return to this same menu and validate the filter without specifying a route name.



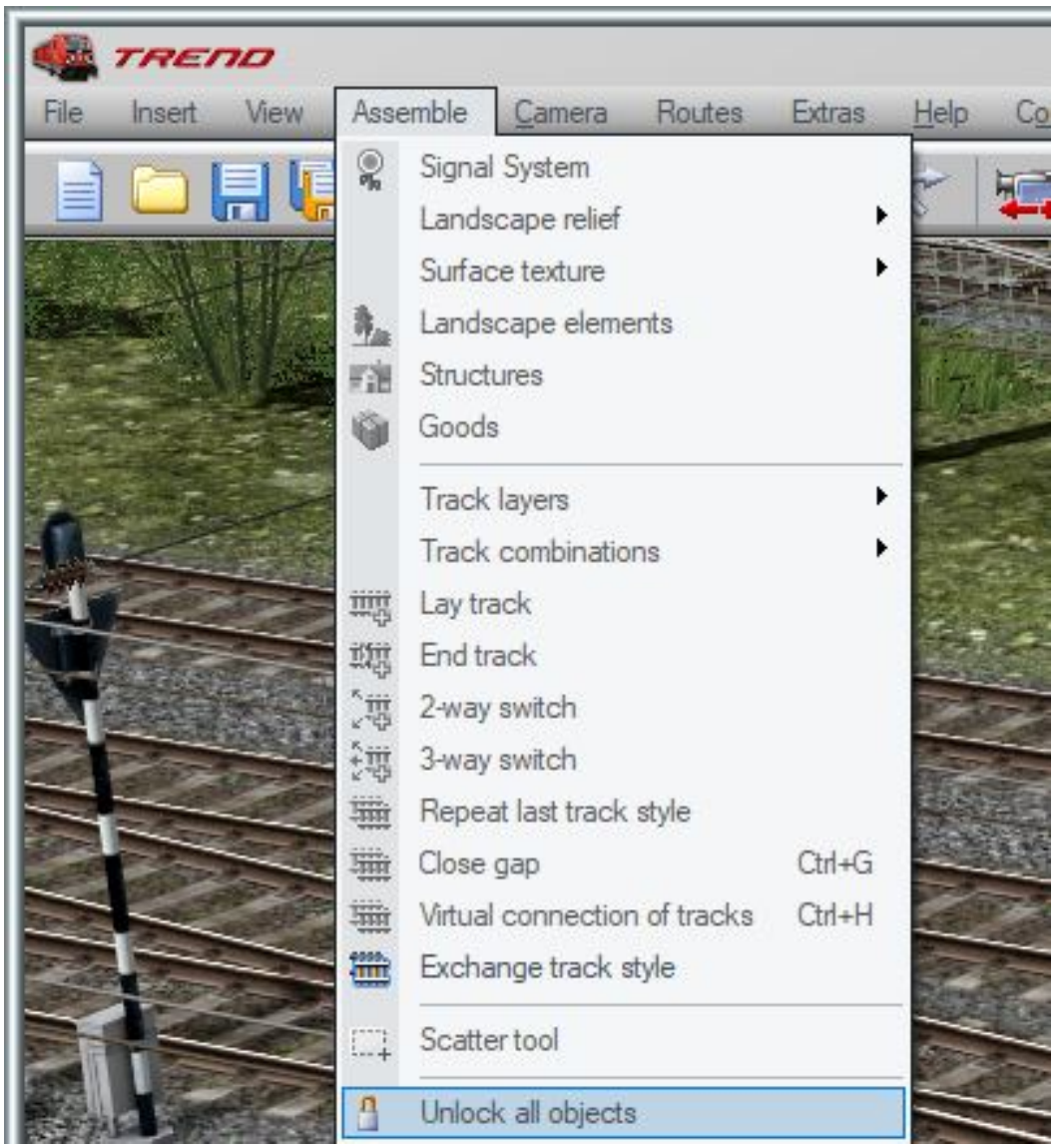
This interesting new feature, which is based on demands from EEP users, could once again be created and added.

Management of the entire train in manual mode

Plug-in 1 for EEP 16.1 allows you to manage the entire train in manual mode. To activate this function from the control window, tick the field marked in red and select "train set". In this way, your action, for example releasing the brake, no longer affects only one rail car, but the whole train.



Complete unlocking of all locked items.



At the repeated request of users, Plug-in 1 for EEP 16.1 finally provides the possibility to unlock all locked objects with a centralized action. This function can be found in the "Assemble" menu

Important information:

This feature does not affect rails with locked ends. It can be used either by going to the menu indicated above or by using the key combination Ctrl+Shift+Right-click on the concerned object

30 new LUA functions

Plug-in 1 for EEP 16.1 comes with 30 new LUA functions:

EEPActivateCtrlDesk()		**EEPActivateCtrlDesk(#Name)**
Type:	Function	OK = EEPActivateCtrlDesk("#120_Stellpult")
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Calls up the control desk in the radar window	
Notes:	• Argument is the complete control desk name as a string • Return value is true when the execution was successful, else false.	

EEPRollingstockSetHorn()		**EEPRollingstockSetHorn("#Name")**
Type:	Function	OK = EEPRollingstockSetHorn("#Car")
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Turns the warning sound of the specified rolling stock on or off.	
Notes:	• First argument is the complete rolling stock name as a string • Second argument is either true (= to sound the horn) or false (= to turn it off). • Return value is true when the execution was successful, else false.	

EEPRollingstockSetHook()		EEPRollingstockSetHook("#Name" , true|false)
Type:	Function	OK = EEPRollingstockSetHook("#Crane truck", true)
Caller:	Script	
Defined in:	EEP	
Parameters:	two	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Activates or deactivates the merchandise hook of the specified rolling stock.	
Notes:	• First argument is the complete rolling stock name as a string • Second argument is either true (= hook) or false (= unhook). • Return value is true when the execution was successful, else false.	

EEPRollingstockGetHook()		EEPRollingstockGetHook("#Name")
Type:	Function	OK, Statut = EEPRollingstockGetHook("#Crane truck")
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Indicates whether or not the merchandise hook of a specific rolling stock is activated	
Notes:	• Argument is the complete rolling stock as a string • First return value is true when the execution was successful, else false. • Second return value indicates whether the hook is deactivated = 0, activated = 1, currently in office = 3	

EEPRollingstockSetHookGlue()		**EEPRollingstockSetHookGlue("#Name", true|false)**
Type:	Function	OK = EEPRollingstockSetHookGlue("#Crane truck", true)
Caller:	Script	
Defined in:	EEP	
Parameters:	two	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Influences the behaviour of goods on a rolling stock's hook	
Notes:	• First argument is the complete rolling stock name as a string • Second argument is either true (= hook glue activated) or false (= hook glue deactivated). • Return value is true when the execution was successful, else false.	

EEPRollingstockGetHookGlue()		**EEPRollingstockGetHookGlue("#Name")**
Type:	Function	OK = EEPRollingstockGetHookGlue("#Crane truck")
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Enquires the behaviour of goods on the rolling stock's hook	
Notes:	• Argument is the complete rolling stock name as a string • Return value indicates whether the hook glue is deactivated = 0, activated = 1, currently in office = 3	

EEPRollingstockGetMileage()		EEPRollingstockGetMileage("#Name")
Type:	Function	OK, Distance = EEPRollingstockGetMileage(“#Car”)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on the rolling stock’s covered distance	
Notes:	• Argument is the complete rolling stock name as a string • First return value is true when the execution was successful, else false. • Second return value indicates the distance in meters that the rolling stock has traveled since its use in EEP	

EEPRollingstockGetPosition()		EEPRollingstockGetPosition("#Name")
Type:	Function	OK, PosX, PosY, PosZ = EEPRollingstockGetPosition(“#Car”)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	four	
Requires:	EEP16.1	
Purpose:	Enquires the rolling stock’s position in the EEP coordinate system	
Notes:	• Argument is the complete rolling stock name as a string • First return value is true when the execution was successful, else false. • Second return value indicates its X position in meters • Third return value indicates its Y position in meters • Forth return value indicates its Z position in meters	

EEPRollingstockSetUserCamera()		EEPRollingstockSetUserCamera("#Name",X, Y, Z, Rot X, RotY)
Type:	Function	OK = EEPRollingstockSetUserCamera(“#Car”, 3, 4, 5, 30, 45)
Caller:	Script	
Defined in:	EEP	
Parameters:	six	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets the user-defined tracking camera (called up with the 9 key)	
Notes:	• First argument is the complete rolling stock name as a string • From second to sixth argument the X, Y, Z position and the Rot X and Rot Y orientation of the camera is set in relation to the defined car • Return value is true when the execution was successful, else false.	

EEPGetCameraPosition()		EEPGetCameraPosition()
Type:	Function	OK, PosX, PosY, PosZ = EEPGetCameraPosition()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	four	
Requires:	EEP16.1	
Purpose:	Enquires the camera’s current position	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the camera’s X position in meters • Third return value is the camera’s Y position in meters • Fourth return value is the camera’s Z position in meters	

EEPGetCameraRotation()		EEPGetCameraRotation()
Type:	Function	OK, RotX, RotY, RotZ = EEPGetCameraRotation()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	four	
Requires:	EEP16.1	
Purpose:	Enquires the camera’s current orientation	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the camera’s X orientation in degrees • Third return value is the camera’s Y orientation in degrees • Fourth return value is the camera’s Z orientation in degrees	

EEPSetCameraPosition()		EEPSetCameraPosition()
Type:	Function	OK = EEPSetCameraPosition(3, 4, 5)
Caller:	Script	
Defined in:	EEP	
Parameters:	three	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets the camera’s position	
Notes:	• First argument sets the camera’s X position in meters • Second argument sets the camera's Y position in meters • Third argument sets the camera's Z position in meters • Return value is true when the execution was successful, else false.	

EEPSetCameraRotation()		EEPSetCameraRotation()
Type:	Function	OK = EEPSetCameraRotation(30, 45, 45)
Caller:	Script	
Defined in:	EEP	
Parameters:	four	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets the camera's orientation	
Notes:	• First argument sets the camera's X orientation • Second argument sets the camera's Y orientation • Third argument sets the camera's Z orientation • Return value is true when the execution was successful, else false .	

EEPRollingstockGetSmoke()		EEPRollingstockGetSmoke("#Name")
Type:	Function	OK, fumee = EEPRollingstockGetSmoke("#Car")
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Enquires if the smoke of a rolling stock is currently turned on or off.	
Notes:	• Argument is the complete rolling stock name as a string • First return value is true when the execution was successful, else false. • Second return value is either 0 = the smoke is turned off or 1 = the smoke is turned on.	

EEPRollingstockSetSmoke()		EEPRollingstockSetSmoke("#Name", true | false)
Type:	Function	OK = EEPRollingstockSetSmoke(“#Car”, true)
Caller:	Script	
Defined in:	EEP	
Parameters:	two	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Turns the rolling stock’s smoke on or off	
Notes:	• First argument is the complete rolling stock name as a string • Second argument is either true = turn on or false = turn off the smoke. • Return value is true when the execution was successful, else false.	

EEPGoodsGetRotation()		EEPGoodsGetRotation("#Name")
Type:	Function	OK = EEPGoodsGetRotation(“#Container”)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	four	
Requires:	EEP16.1	
Purpose:	Enquires the goods’ orientation	
Notes:	• Argument is the complete goods name as a string • First return value is true when the execution was successful, else false. • Second return value is the goods’ X orientation in degrees. • Third return value is the goods’ Y orientation in degrees. • Fourth return value is the goods’ Z orientation in degrees.	

EEPStructureGetRotation()		EEPStructureGetRotation("#Name")
Type:	Function	OK, RotX, RotY, RotZ = EEPStructureGetRotation("#Tunnel")
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	four	
Requires:	EEP16.1	
Purpose:	Enquires the structure's or landscape element's orientation	
Notes:	• Argument is the complete structure's or landscape element's name as a string • First return value is true when the execution was successful, else false. • Second return value is the structure's or landscape element's X orientation in degrees. • Third return value is the structure's or landscape element's Y orientation in degrees. • Fourth return value is the structure's or landscape element's Z orientation in degrees.	

EEPGetWindIntensity()		EEPGetWindIntensity()
Type:	Function	OK, wind-intensity = EEPGetWindIntensity()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on wind intensity	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the wind intensity in percent	

EEPGetRainIntensity()		EEPGetRainIntensity()
Type:	Function	OK, rain-intensity = EEPGetRainIntensity()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on rain intensity	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the rain intensity in percent	

EEPGetSnowIntensity()		EEPGetSnowIntensity()
Type:	Function	OK, snow-intensity = EEPGetSnowIntensity()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on snow intensity	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the snow intensity in percent	

EEPGetHailIntensity()		EEPGetHailIntensity()
Type:	Function	OK, hail-intensity = EEPGetHailIntensity()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on hail intensity	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the hail intensity in percent	

EEPGetFogIntensity()		EEPGetFogIntensity()
Type:	Function	OK, fog-intensity = EEPGetFogIntensity()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on fog intensity	
Notes:	• First return value is true when the execution was successful, else false • Second return value is the fog intensity in percent	

EEPGetCloudIntensity()		EEPGetCloudIntensity()
Type:	Function	OK, cloud-intensity = EEPGetCloudIntensity()
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	two	
Requires:	EEP16.1	
Purpose:	Provides information on cloud intensity	
Notes:	• First return value is true when the execution was successful, else false. • Second return value is the cloud intensity in percent	

EEPOnSaveAnl()		EEPOnSaveAnl(saving path)
Type:	Function	function EEPOnSaveAnl(chemin-d-enregistrement) print (“Le projet “..chemin-d-enregistrement..“a été enregistrée!” end
Caller:	Script	
Defined in:	EEP	
Parameters:	none	
Returns:	one	
Requires:	EEP16.1	
Purpose:	EEP automatically calls up this function when saving the project. In the script we set this function and specify what there is to do when saving.	
Notes:	• Argument is the project’s saving path including the project’s file name as a string. The variable in quotation marks takes this value into account for future use. • EEP does not expect any feedback when using this function.	

EEPSetWindIntensity()		EEPSetWindIntensity(wind-intensity)
Type:	Function	OK = EEPSetWindIntensity(100)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets wind intensity	
Notes:	• Argument is wind intensity in percent. • Return value is true when the execution was successful, else false.	

EEPSetRainIntensity()		EEPSetRainIntensity(rain-intensity)
Type:	Function	OK = EEPSetRainIntensity(50)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets rain intensity	
Notes:	• Argument is rain intensity in percent • Return value is true when the execution was successful, else false.	

EEPSetSnowIntensity()		EEPSetSnowIntensity(snow-intensity)
Type:	Function	Ok = EEPSetSnowIntensity(50)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets snow intensity	
Notes:	• Argument is snow intensity in percent • Return value is true when the execution was successful, else false.	

EEPSetHailIntensity()		EEPSetHailIntensity(hail-intensity)
Type:	Function	OK = EEPSetHailIntensity(100)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets hail intensity	
Notes:	• Argument is hail intensity in percent • Return value is true when the execution was successful, else false.	

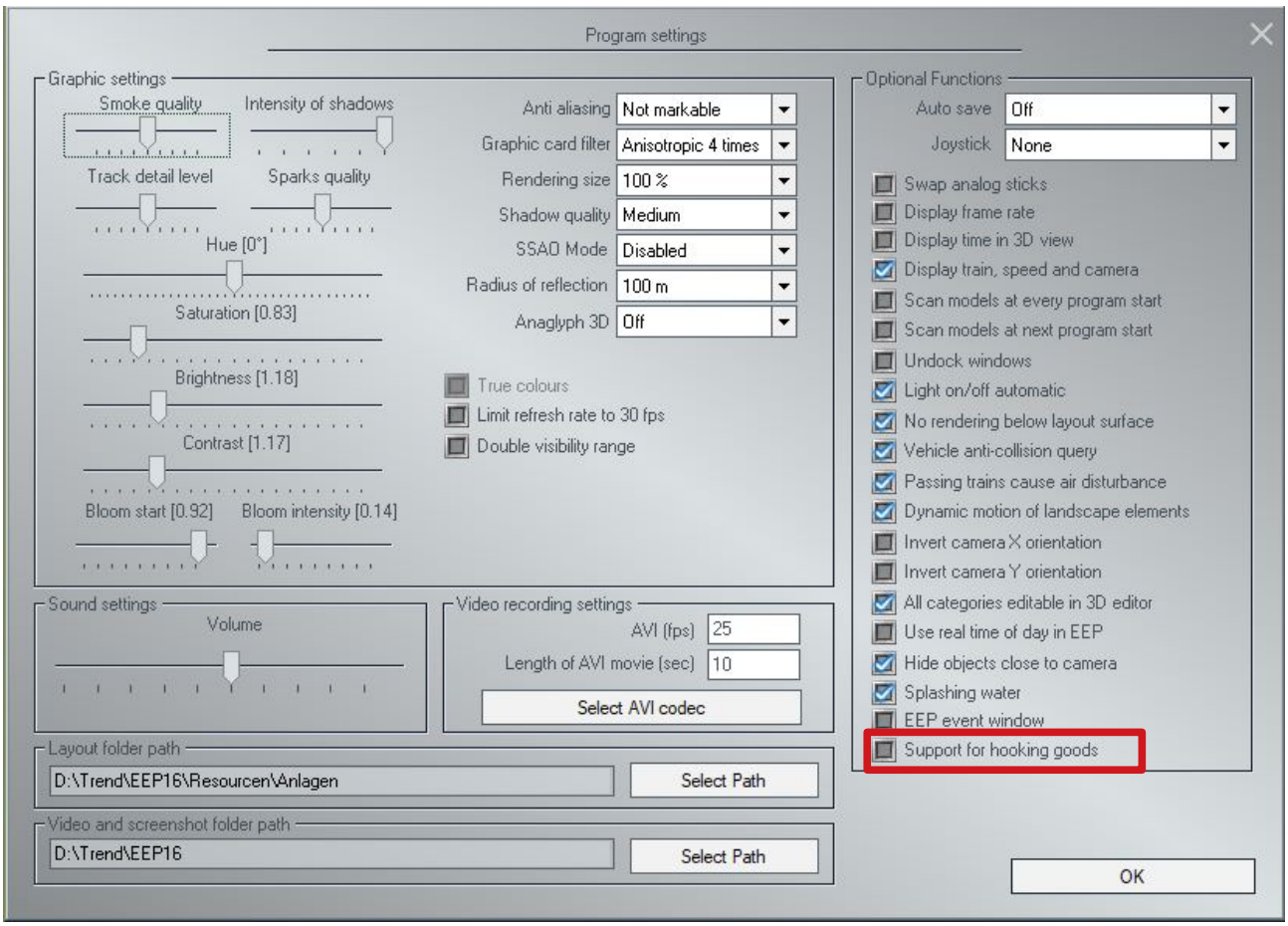
EEPSetFogIntensity()		EEPSetFogIntensity(fog-intensity)
Type:	Function	OK = EEPSetFogIntensity(100)
Caller:	Script	
Defined in:	EEP	
Parameters:	one	
Returns:	one	
Requires:	EEP16.1	
Purpose:	Sets fog intensity	
Notes:	• Argument is fog intensity in percent • Return value is true when the execution was successful, else false.	

EEPSetCloudIntensity()		EEPSetCloudIntensity(cloud-intensity)
Type:	Function	OK = EEPSetCloudIntensity(100)
Caller:	Script	
Defined in:	EEP	
Parameters:	un	
Returns:	une	
Requires:	EEP16.1	
Purpose:	Sets cloud intensity	
Notes:	• Argument is cloud intensity in percent • Return value is true when the execution was successful, else false.	

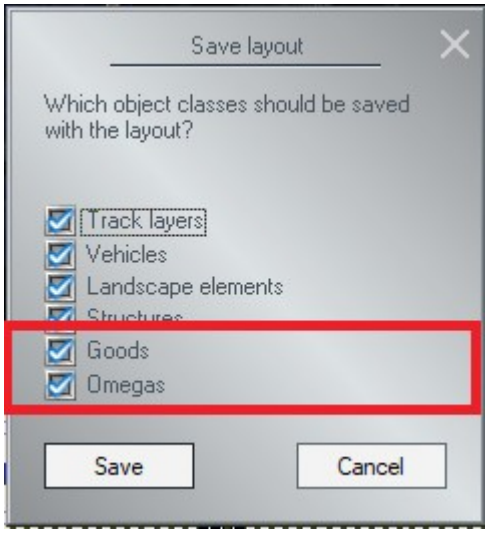
”Magnetic” interlocking function while loading goods

With Plug-in 1 for EEP 16.1, we have provided vehicles and loading equipment with additional functionality. During loading, the goods specifically designed for this purpose snap firmly into the hook and are thus protected against unintentional movement or slipping. This illustrates EEP's willingness to adapt to Nvidia's PhysX® Open Source engine.

Therefore, a new option has been added to the program parameters, allowing an easier loading of the goods.



The save as function has been extended.



The save as function has been improved with the release of Plug-in 1 for EEP 16.1.

By now it's possible to save goods and Omegas with your project.

Closing remarks:

With the design of Plug-in 1 for EEP 16.1 many wishes of experienced EEP users have been taken into account. The new features make it easier to design and manage your installation. The extension of LUA functions allows you to further automate your installation.

We hope that this first Plug-in for EEP 16.1 will bring you a lot of pleasure.

Your Trend EEP team.