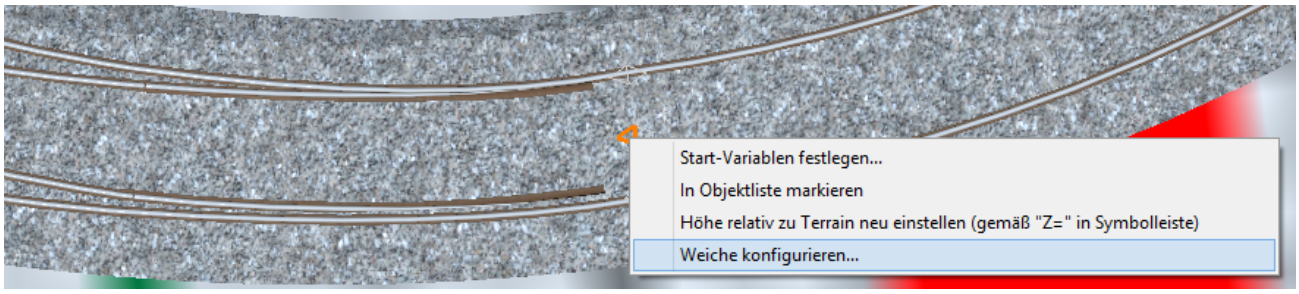
Routes, signals and electrical switches

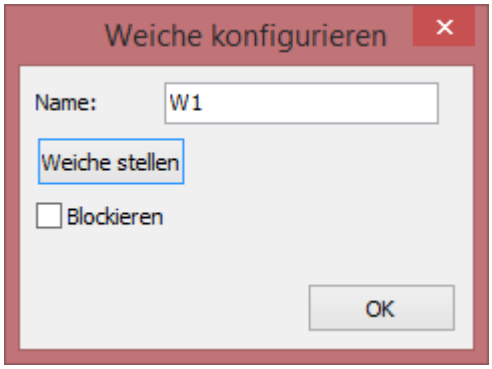
Table Of Contents

1 Prepare switches

It is not *necessary*, but *advisable*, to give names to the switches (at least those which are to be used in routes). To do this in "Track" mode, right-click on the marker pointing in the direction of the turnout tip and click on "Configure turnout..." in the context menu.



In the appearing dialog box the name of the switch can be entered immediately.

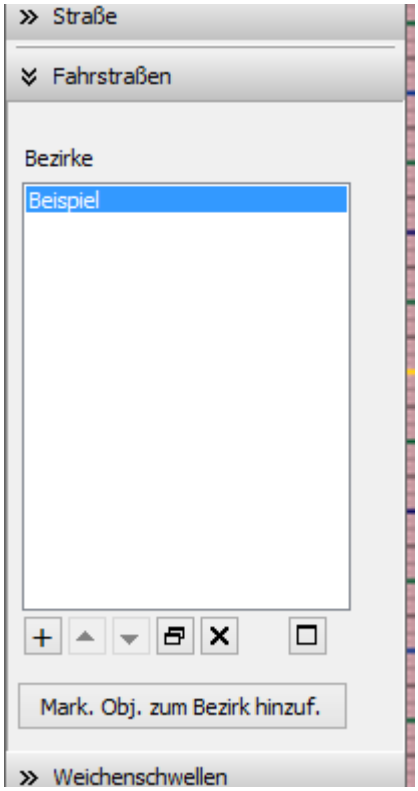


2 Create district

For the sake of clarity and performance, routes are grouped into districts that work independently of each other. These can be imagined as individual interlockings.

The grouping is particularly relevant for performance because each district contains an exclusion matrix which has the size "number of routes to square" and which is used to check whether an "enemy" route is already in use when a route is requested. Suppose you had a large network with 20 signal boxes with 10 routes each. If you didn't define districts, you would need a 200x200 matrix with 40,000 entries. If the routes were grouped to the respective signal boxes, then there would be 20 matrices with only 10x10 = 100 fields, i.e. only a total of 1,000 entries (compared to 40,000!).

The districts are managed via the list box known in many other areas, which is located on the left in the section "Routes".

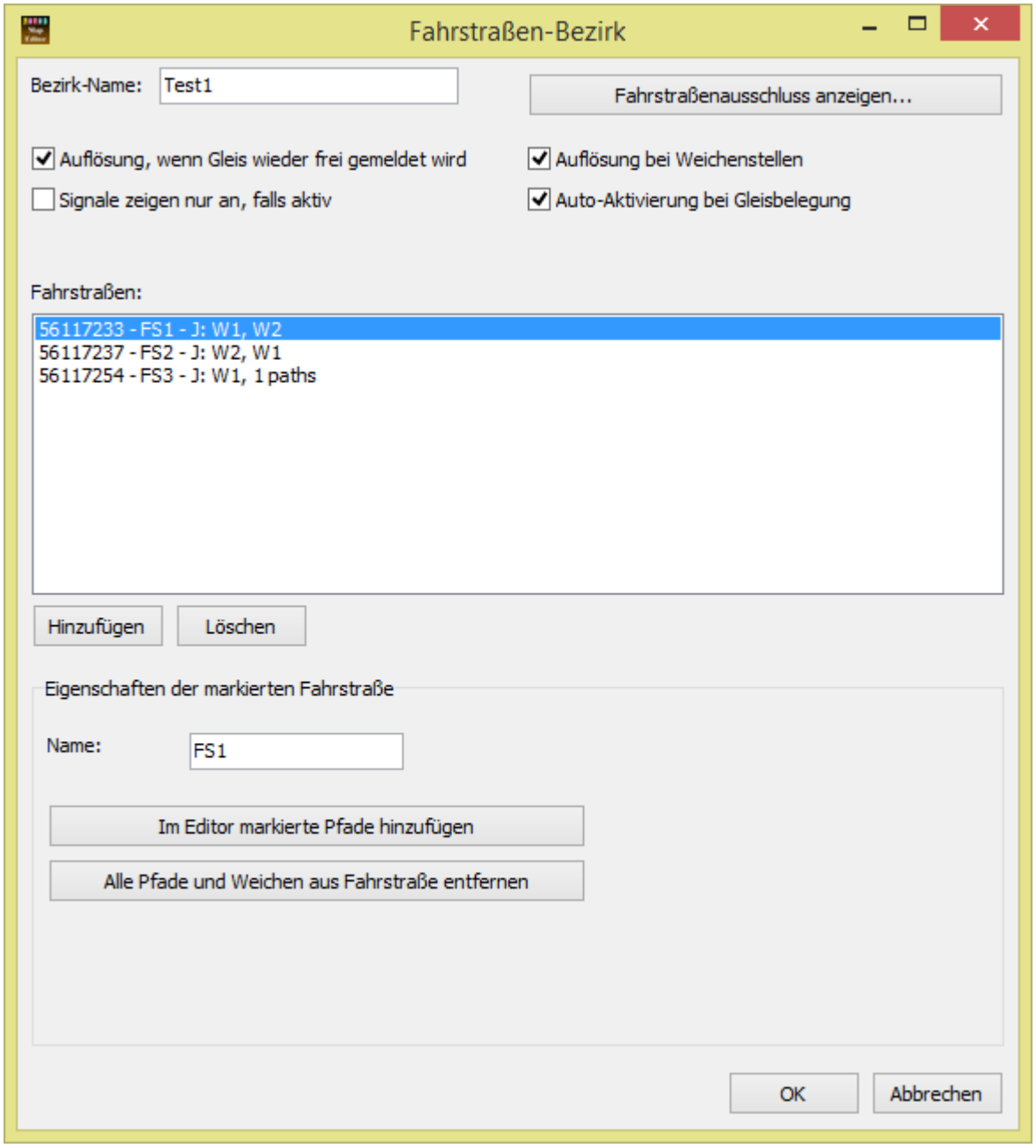


3 Create route

In the Properties dialog box (accessible via double-click or the button with the rectangle below the list) the district can first be given a name and further basic properties of the district can be set:

- **Deactivate, if track is free again**: Active routes will be activated automatically, if its tracks are free after they were occupied before. Option is set on for default.
- **Signals show only, if route is active:** If this option is set off, the switch signals assigned to a special route will be also on, if the route *could* be activated, even if it is not activated. So they will always show the current direction of the junction and not only if there is an assigned route active. Option is set on for default.
- **Deactivation by switching junction:** If you switch a junction manually although there is a route active using this switch, it will be automatically deactivated. Option is set on for default.
- **Auto activation by track occupation:** if a route will be occopied even if it is not active and if the junction directions fit and there is no "enemy" route active, it should be activated automatically. Option is set on for default.

Below these options, you can create or remove the individual routes with "Add" and "Delete". If one of them is clicked, its properties can be edited directly below the list:

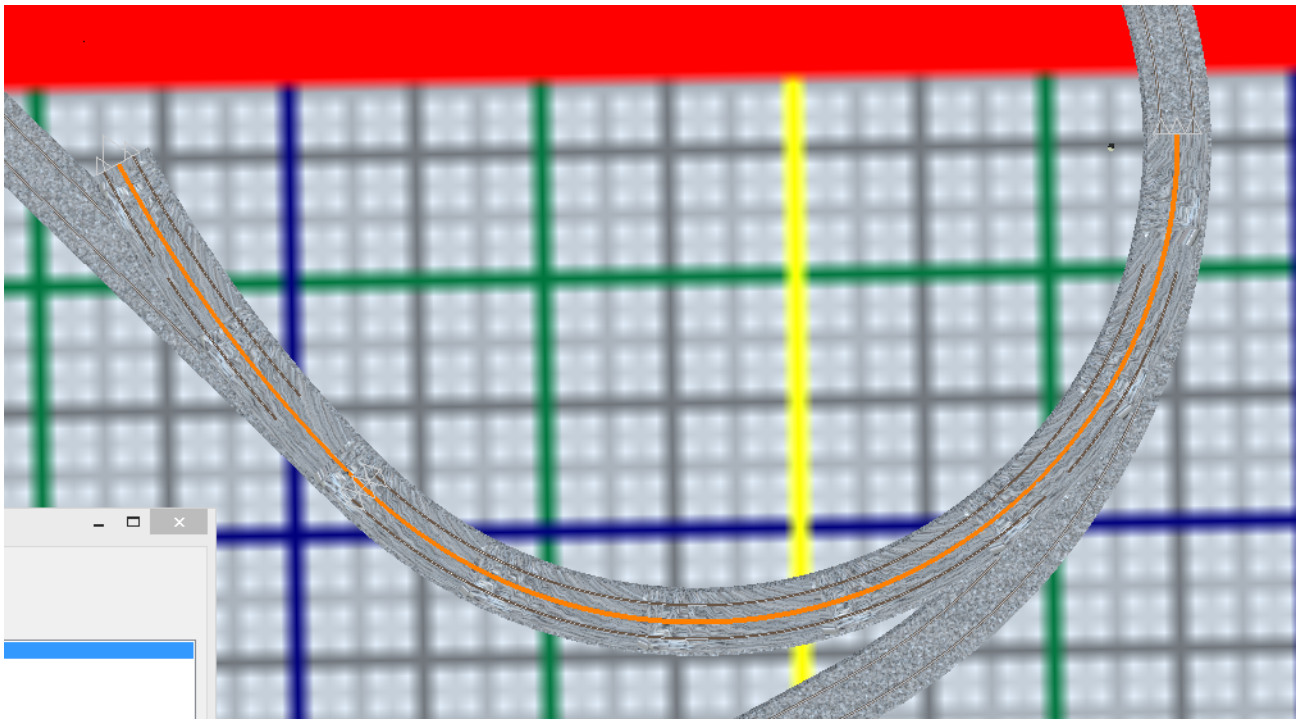


This currently includes the name of the individual route and the switches that are to be set when the route is activated.

3.1 Adding switches and paths

Although the window is open, the MapEditor can still be used in the background. The following steps can be carried out before opening the dialog box, but also if it is already open:

First you have to switch to the "Track" mode. Then use [Shift] to select all paths along which the route runs:



Finally, you only have to create/select the desired route and click on "Add paths marked in the editor". The paths will then be saved and the marker will automatically determine which points have to be set to make this route possible.

As soon as the paths and switches have been imported, this can be checked by checking the entry of the route in the list to see if everything worked out - the switches to be crossed are noted with their names there.

3.2 Deleting switches

The switches are deleted simply by clicking on "Remove all paths and switches from the route".

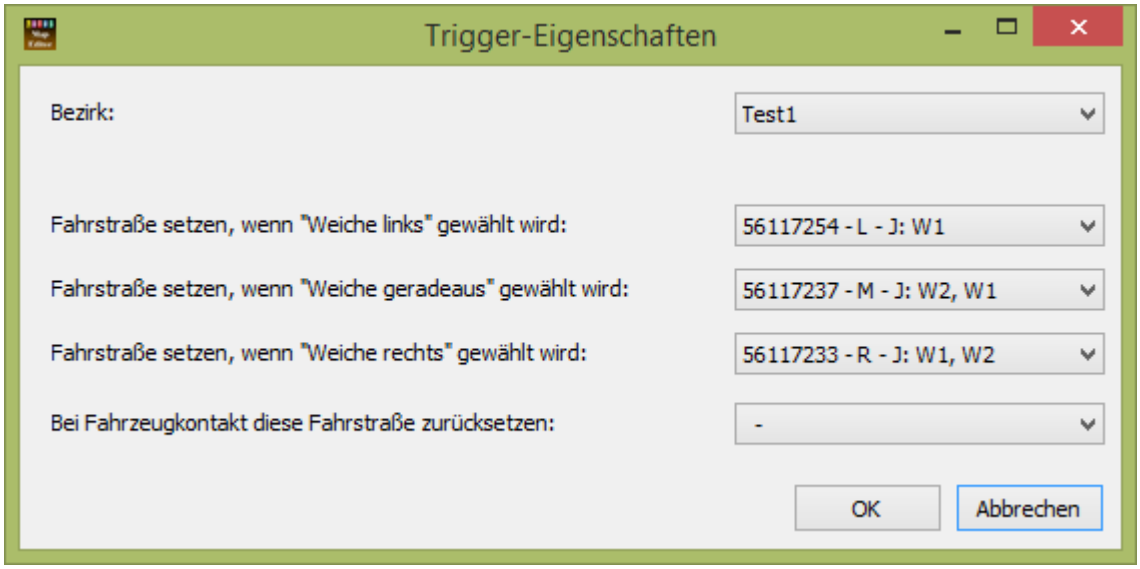
4 Configuring triggers and beacons

Now that the routes have been defined, it is time to use triggers to activate and deactivate them automatically. "Triggers" are special scenery objects (invisible in the simulation). Triggers will be able to take over various tasks, but at the moment they are only used to insert and resolve routes.

4.1 Placing Triggers

You can find the trigger object on the right in the category "Helpers". It just has to be placed "approximately" on the track; the internal "connection" to the track is still guaranteed. An exact alignment/rotation is also not necessary, but the rough direction (forwards or backwards) is very important, because the sensors/triggers always only react to the sensors/triggers of "their" direction.

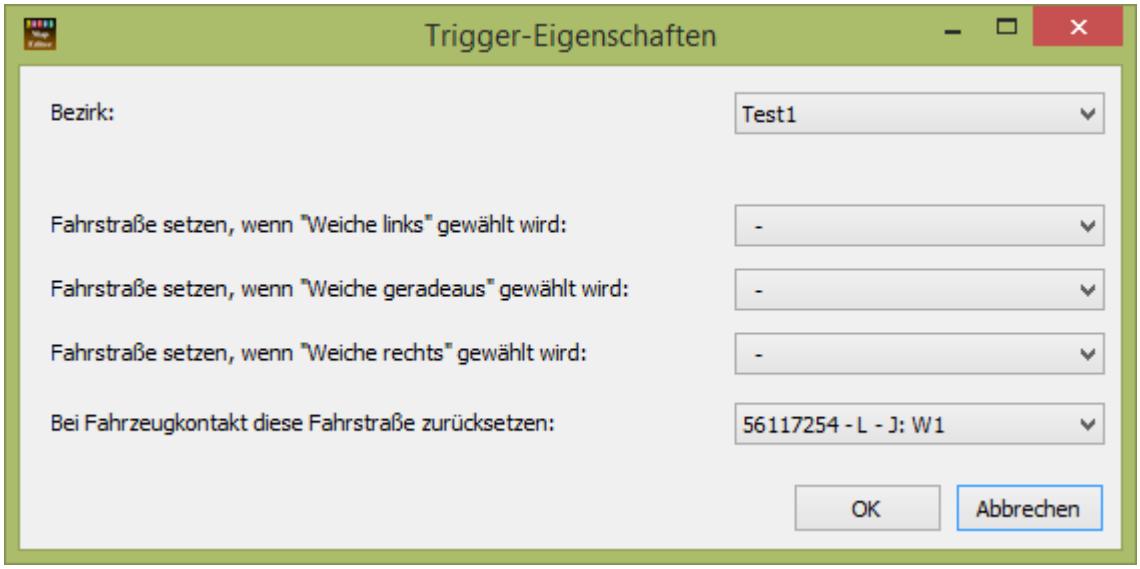
Right-click on the trigger and select "Trigger Properties..." to open the properties dialog:



First, you select the route district to which the trigger should belong.

If the vehicle sends a positioning command to the trigger, then it should insert a certain route - depending on the positioning command. This behaviour can be set in the following three dropdown menus. If the trigger is to ignore certain or all positioning commands, " - " must be selected in each case.

If the vehicle leaves the area of the driveway, it is to be dissolved (deactivated) again. If this should not be done by the track clearance messages, a trigger can also be assigned a route for this purpose. Here the trigger does not "wait" for a certain information/request from the vehicle, but the cancellation is triggered immediately when the trigger is "touched".



5 Connecting simple signals

A simplified system is used for simple signals (e.g. standard BOStrab switch signals such as W1, W2, W3 etc.), where a scenery object only indicates whether a certain route has been taken or not:

- Placing and selecting the signal
- Select the district on the left
- Click "Add marked object to selected district" below the district list.
- Select the route to display this signal.

6 More complex signals

6.1 Signalobjekt konfigurieren

Komplexe Signale müssen zuerst vorbereitet werden: Im Object & Vehicle Tool gibt es im Bereich "Allgemeine Einstellungen" die Schaltfläche "Signalbegriffe". Hierüber kann ein Texteingabe-Feld geöffnet werden, in welches (vereinfacht gesagt) diejenigen Signalbegriffe eingetragen werden müssen, die das Signal zeigen kann. Hierbei ist aber zu beachten, dass

- das "Nichtvorhandensein" von Fahrstraßen usw., also ein Hp0 oder ein dunkles Weichensignal usw., nicht ein eigener Signalbegriff sind
- keine "Kombinationen" eingetragen werden
- maximal sind 32 Zeilen und somit Begriffe erlaubt

Kann das Signal beispielsweise sowohl Vor- als auch Hauptsignal-Begriffe anzeigen, dann erhält es bspw. die Einträge:

- Hp1
- Hp2
- Vr1
- Vr2

FALSCH wäre also sowas wie:

- ~~Hp0+Vr0~~
- ~~Hp1+Vr0~~
- ~~Hp2+Vr0~~

- ~~Hp1+Vr1~~
- ~~usw.~~

Warum das so ist, wird vermutlich klarer, wenn wir uns mit der Konfiguration im MapEditor beschäftigt haben!

6.2 Signal im MapEditor konfigurieren

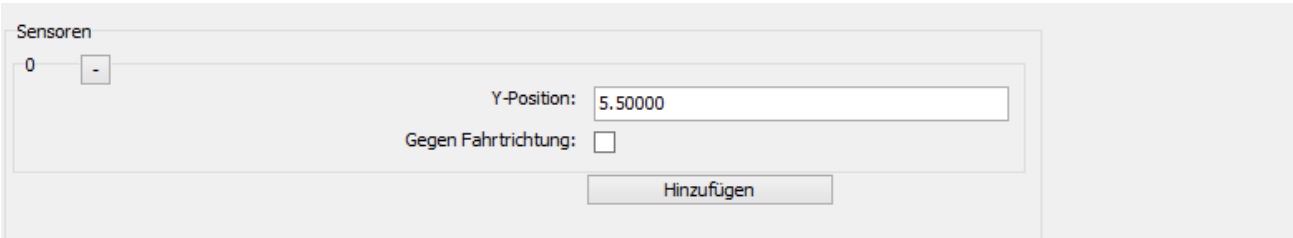
Sobald dem Signal-Objekt auf die beschriebene Weise die möglichen Signalbegriffe mitgeteilt wurden, erscheint im MapEditor, wenn man auf das Signal mit rechts klickt, im Kontextmenü der Eintrag "Signal-Eigenschaften". Wird dieser angeklickt, öffnet sich das hierfür vorgesehene Dialogfeld.

Von der Struktur her gibt es für jeden Signalbegriff einen Listen-Abschnitt, zu dem jeweils beliebig viele Fahrstraßen hinzugefügt werden können. Die Anzahl der Signalbegriffe ergibt sich durch die Eingaben beim Signal-Objekt im Object & Vehicle Tool. Für jeden Fahrstraßeneintrag kann jeweils individuell der Bezirk ausgewählt werden, damit es auch möglich ist, Signalbegriffe bezirksübergreifend zusammen zu stellen. Außerdem wird jeweils ausgewählt, ob geprüft werden soll, ob die Fahrstraße aktiv ist, deren Gleise belegt sind oder ob sie aktiv ist und gleichzeitig ihre Gleise nicht belegt sind.

Das ganze System arbeitet nun wie folgt: Für jeden Signalbegriff wird stets geprüft, ob für zumindest eine der Fahrstraßen die ausgewählte Bedingung erfüllt ist. Falls dies der Fall ist, "gilt" dieser Signalbegriff. Schließlich werden die so ermittelten "Gesamtzustände" jeder Fahrstraße kodiert und per Script an das Signal übertragen.

7 Equipping the vehicle

For a vehicle to be able to interact with the beacons, it must have at least one sensor, bi-directional vehicles require at least one at each end. This is set up in the object settings in the "Sensors" section:



The parameters mean:

- Y position: The position of the sensor on the vehicle in the longitudinal direction.
- Against direction of travel: The sensors always only interact with triggers that have the same orientation, so it must be specified here in which direction this sensor is to be installed.

8 Script Implementation

8.1 Vehicle

On the script side, there are already the following possibilities of interaction between vehicle sensor and

track trigger:

- Send messages to all triggers in range with `SendMessageToTrigger(Self; ID: string; value: string; sensor_index: integer);`. So far the ID "SWITCH" is supported with the values (each as string!) "0" (left), "1" (right) and "2" (straight ahead) for transmitting a positioning command.
- If a procedure is included in the script with the declaration procedure `OnEnterLeaveTrigger(triggerid: string; entering: boolean; sensorindex: integer);`, it is always called when the vehicle reaches or leaves a trigger. "Triggerid" is currently not used, later it will be used e.g. for identification for IBIS etc..

In GT6N, for example, both functions are used together: When the driver activates a positioning command, this value (for a certain maximum distance) is "earmarked". If OnEnterLeaveTrigger is then triggered, the script immediately uses SendMessageToTrigger and sends the command to the trigger. Other variants are also possible: If the trigger has a larger radius (e.g. for contact positioning commands or longer contact loops), SendMessageToTrigger could also be triggered immediately if the driver actuates the positioning command.

8.2 Simple Signals

Similar to the traffic lights, only the "PUBLIC" variable `signalroute_active: boolean;` must be included. If everything is linked correctly and the route is inserted, this variable becomes "true", otherwise it is "false". However, only simple signals can be realized with this variable. Complex signals will be added later!

8.3 Komplexe Signale

Auch bei komplexen Signalen werden die Daten über eine "PUBLIC"-Variable, nämlich `signalstate: integer;` übertragen. Achtung: Im Gegensatz zu einfachen Signalen handelt es sich hierbei um eine Integer-Variable!

In diese werden nun im Betrieb die Signalbegriffe bitweise eingetragen – siehe hierzu auch Bit-Flags erklärt . Verfügt das Signal z.B. über die Begriffe Hp1, Hp2, Hp3 und Hp4, dann wird der Wert der Variable `signalstate` die folgenden Werte annehmen:

- 1, falls *nur* die Hp1-Bedingungen gelten
- 2, falls *nur* die Hp2-Bedingungen gelten
- 4, falls *nur* die Hp3-Bedingungen gelten oder
- 8, falls *nur* die Hp4-Bedingungen gelten.

Gelten mehrere Bedingungen gleichzeitig, dann werden diese Werte einfach zusammengerechnet.

Auseinander nehmen kann man die Bedingungen dann wie folgt:

- wenn `(signalstate and 1) <> 0`, dann ist die Hp1-Bedingung erfüllt
- wenn `(signalstate and 2) <> 0`, dann ist die Hp2-Bedingung erfüllt
- wenn `(signalstate and 4) <> 0`, dann ist die Hp3-Bedingung erfüllt
- wenn `(signalstate and 8) <> 0`, dann ist die Hp4-Bedingung erfüllt

Komplexer, aber trotzdem problemlos ist es, wenn Zusatz-Anzeiger zur Anwendung kommen.

Beispiel: Ein nur Hp4-fähiges U-Bahnsignal mit Gleis-Nummern-Anzeige. Die Signalbegriffe wären dann:

- Hp4
- Gleis 1
- Gleis 2
- Gleis 3
- Gleis 4
- Gleis 5
- Gleis 6
- Gleis 7
- Gleis 8
- Gleis 9

Angenommen, es gibt drei Fahrstraßen A, B und C, die auf Gleis 3, 4 und 7 führen.

Dann werden im MapEditor in der Signal-Konfiguration dem Signalbegriff *Hp4* alle drei Fahrstraßen zugeordnet, den Signalbetriffen *Gleis 3*, *Gleis 4* und *Gleis 7* jeweils nur die Fahrstraße A, B und C, und den restlichen Signalbegriffen werden gar keine Fahrstraßen zugeordnet.

Nach obiger Formel wird die Variable `signalstate` dann folgende Werte annehmen:

- Fahrstraße A: Hp4+Gleis 3, d.h. 1 + 8 = 9
- Fahrstraße B: Hp4+Gleis 4, d.h. 1 + 16 = 17
- Fahrstraße C: Hp4+Gleis 7, d.h. 1 + 128 = 129

Im Signalscript können dann die Signalbegriffe wieder mit den obigen "and-Formeln" auseinander genommen werden:

Falls `(signalstate and 1) <> 0`, dann soll das Signal auf Hp4 gehen, sonst auf Hp0. Nur, wenn es Hp4 zeigt, dann soll er die restlichen Werte in `signalstate` prüfen und dann entsprechend daraus ableiten, wie er die jeweiligen Gleisnummern anzeigen soll.