

OMSI 1.00

Technical Manual –

Software Development Kit (SDK)

Part 1: Map Editor

State: February, 13th, 2011

© 2011 Marcel Kuhnt

Content

INTRODUCTION	5
LESSON 1: SCENERY OBJECTS AND SPLINES	6
1.1. Copying An Existing Map	6
1.2. Placing A Scenery Object	12
1.3. Trees	15
1.4. Splines / Streets	18
1.5. Intersections and streets	22
1.6. Save Map	29
1.7. Adding Entry Points and Labeling Objects	29
LESSON 2: ENHANCED OBJECT EDITING	32
2.1. Attach Objects At Splines	32
2.2. Attaching Objects At Objects	34
2.3. Label Objects	36
2.4. Place Traffic Lights	37
2.5. Traffic Rules	44
2.6. Map Priorities	47
LESSON 3: CREATE NEW MAP AND TERRAIN EDITING	48
3.1. Create New Map	49
3.2. Tiles	50
3.3. Background Image	50
3.4. Add Tile	50
3.5. Editing Terrain	52
3.6. Terrain Painting	54
LESSON 4: CREATING A NEW BUS ROUTE	56
4.1. Placing streets and intersections	57

4.2.	Terrain lightmaps & „Envir“ tab	62
4.3.	Adjusting traffic rules	66
4.4.	Placing bus stops	67
LESSON 5: SCHEDULED AI TRAFFIC		73
5.1.	Introduction to configuration files	74
5.2.	Adding a new AI group	76
5.3.	Creating the first scheduled AI trip	77
5.4.	Adding your own bus repaint	97
5.5.	AI groups with specific numbers	104
5.6.	Creating a depot file	107
ANHANG 1: KREUZUNGEN MIT DEN DAZU PASSENDEN SPLINE-TYPEN		111
ANHANG 2: REPAINTER-DATEIEN FÜR SD200 UND SD202		113

Introduction

In the beginning, we tried to create a technical handbook like a dictionary. But with increasing complexity of OMSI we just came aware of the usefulness of explaining every topic step by step, from the easiest to the most difficulty aspect. We thought that it would be much better to create the documentation like a tutorial.

Of course it would have been best to have both a tutorial and also a reference book – but this would have been much too much work for us!

So now you get a step by step documentation – but you can also search for special key words with the search functions!

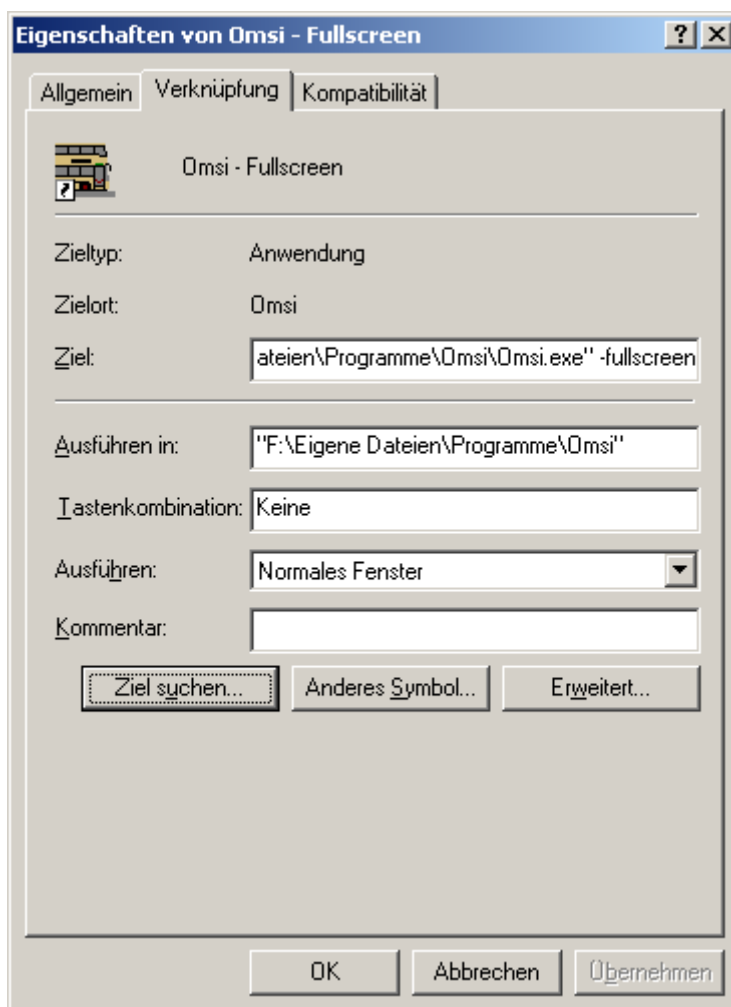
We have so much to say in our technical documentation that there could be some orthographic or grammar mistakes – if you find a mistake, of course you may keep it! 😊

Lesson 1: Scenery Objects and Splines

You will learn the basics about editing maps of OMSI with the map editor. There is no logical limitation of the number of maps in OMSI. Every map is a closed world for itself. You cannot drive from one map to another.

1.1. Copying An Existing Map

To create a copy of a map, open the directory where you have installed OMSI. The fastest way to do so is clicking on the OMSI icon on your desktop with the right mouse button and then on "Properties":



Click on „Search target...” and you will get the OMSI directory.

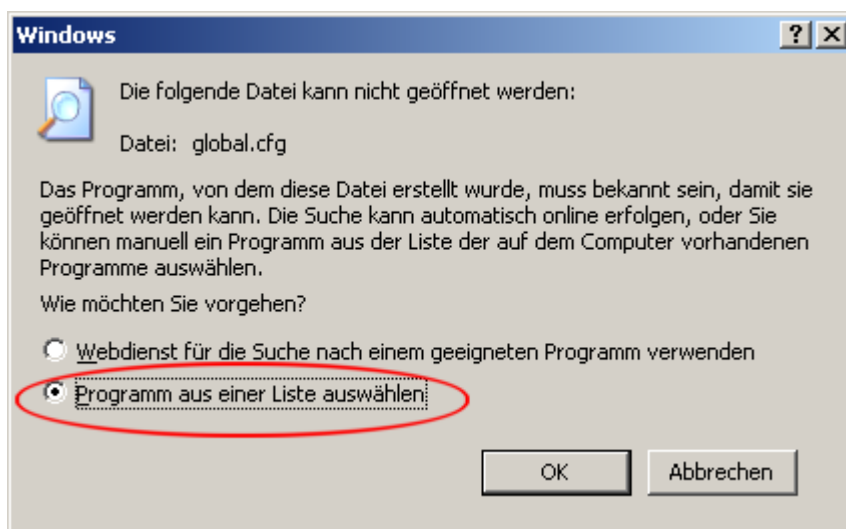
Change into the subdirectory “maps” where you can find all OMSI maps. Copy the directory “Grundorf”. Rename the copy into e.g. „My_Map”.

Now the directory has the right name – but the official name of the map is still „Grundorf“. This will now be changed:

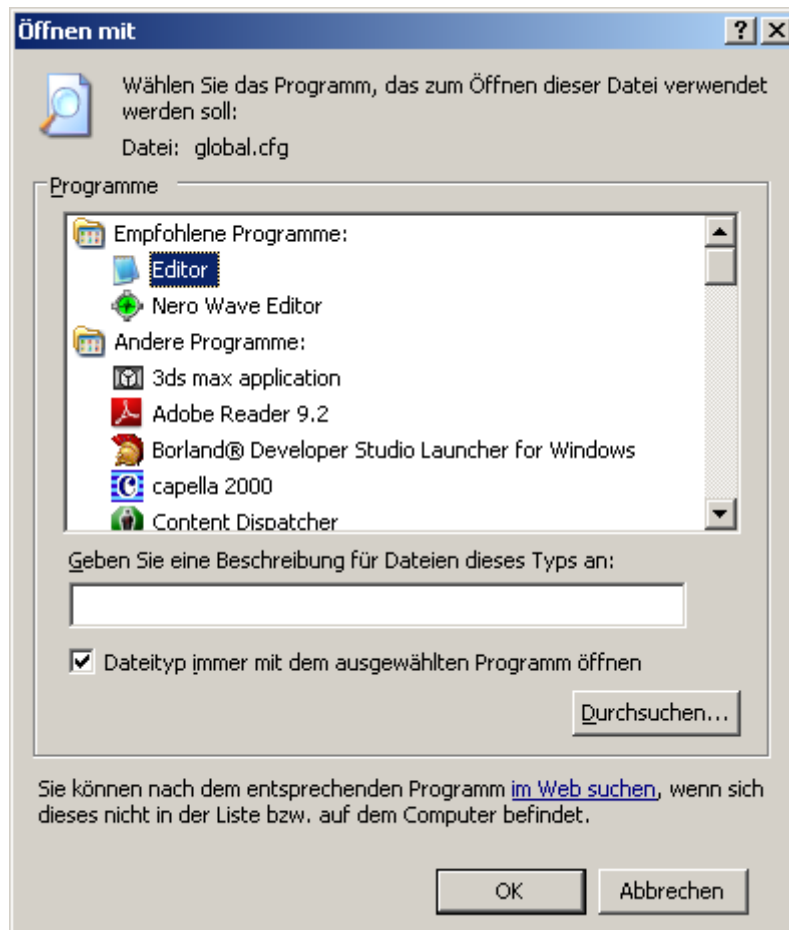
Change into the directory „My_Map“. There you will find all files describing the map. Now, the most interesting file is “global.cfg”. It contains all global properties of the map.

Open this file with the Windows Notepad. Many file types in OMSI may be opened and edited in this program. Since Windows does not know this file type, you first have to tell it to open it with Windows Notepad:

Double click onto the file. The following window pops up where you have to select the marked option and click “OK”:

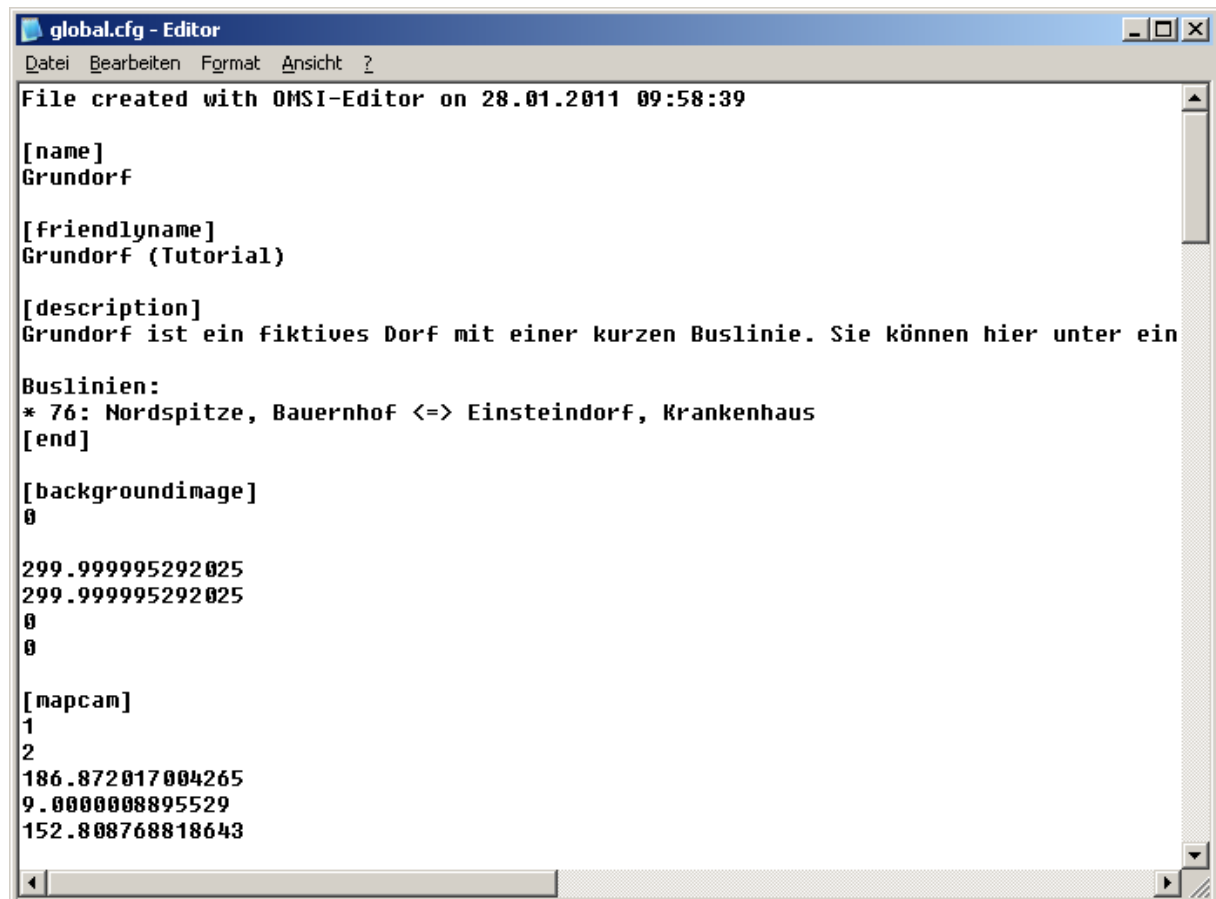


Now the following window pops up:



Select „Notepad“, pay attention, that the box „Open file type every time with this program“ is checked and confirm with “OK”. Now Windows knows that it always has to open this file type with the Windows Notepad.

Now you will see this file window:



```
global.cfg - Editor
Datei Bearbeiten Format Ansicht ?
File created with OMSI-Editor on 28.01.2011 09:58:39

[name]
Grundorf

[friendlyname]
Grundorf (Tutorial)

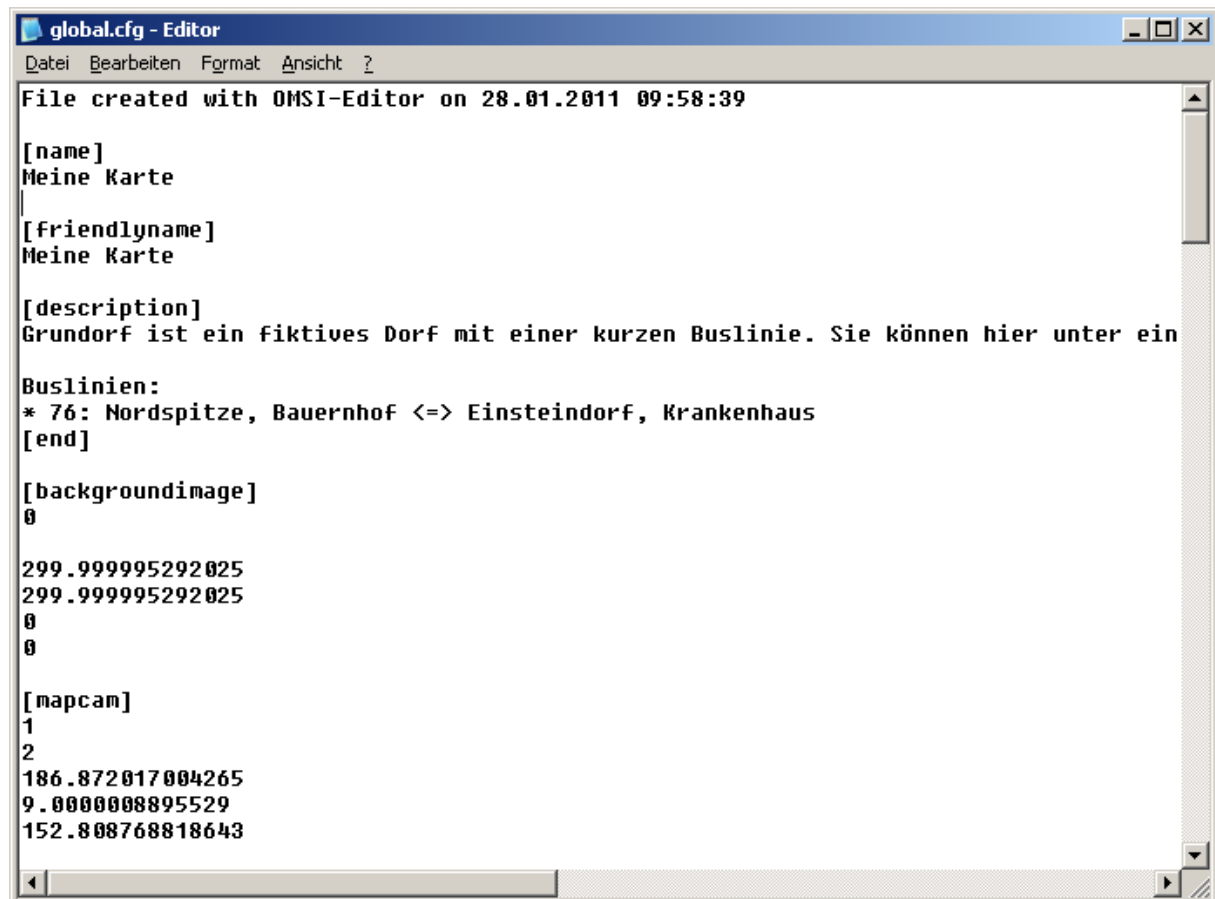
[description]
Grundorf ist ein fiktives Dorf mit einer kurzen Buslinie. Sie können hier unter ein
Buslinien:
* 76: Nordspitze, Bauernhof <=> Einsteindorf, Krankenhaus
[end]

[backgroundimage]
0

299.999995292025
299.999995292025
0
0

[mapcam]
1
2
186.872017004265
9.0000008895529
152.808768818643
```

You can see the keywords „[name]“ and „[friendlyname]“. Please replace in each following line “Grundorf” with “My Map”:

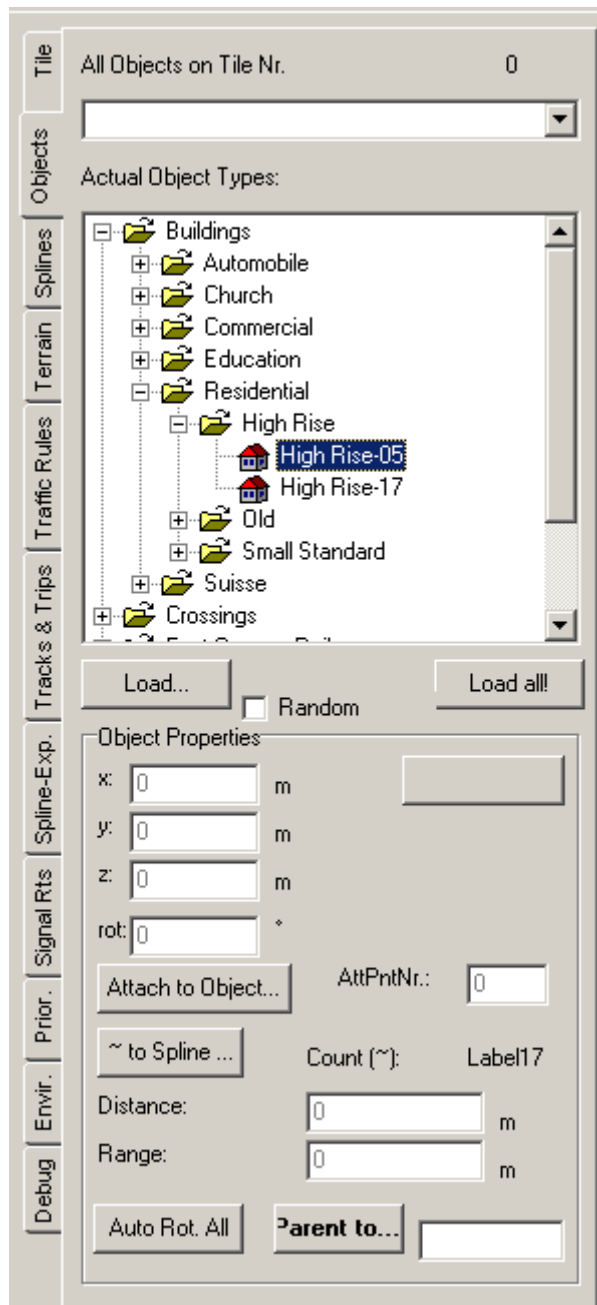


Save and close the file.

Run the OMSI Map Editor. The first question is always "Would you like to continue, where you have stopped last time?". Click "No" to select your new map.

Now you will see the Open Dialog. Open the directory "My_Map" and double click onto "global.cfg".

The Editor now will open your copy of the Grunddorf map, we will now edit in the first lesson.



At the top you can see a drop down field for searching a special object on the current tile. But this function is not necessary in general, but you can try to find “lost” objects.

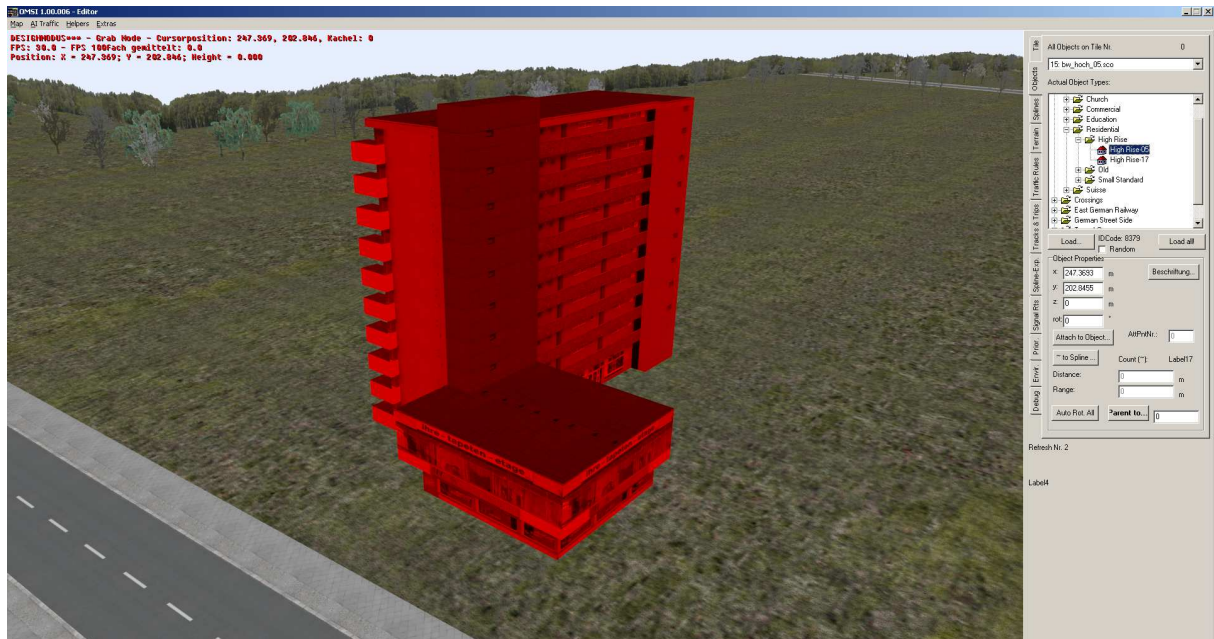
Beneath you find the tree view of scenery object types.

Important: After starting the editor, here you will find exactly these types which are available in the map! But of course you can add objects by clicking on “Load...” (and search for a special object) or by clicking on “Load all!” – then the Editor will add all installed objects into the tree view.

But this is not necessary now, because we will only use objects already added to the tree view.

In the lower area you will find several properties of the current selected object. But there is none selected at the moment.

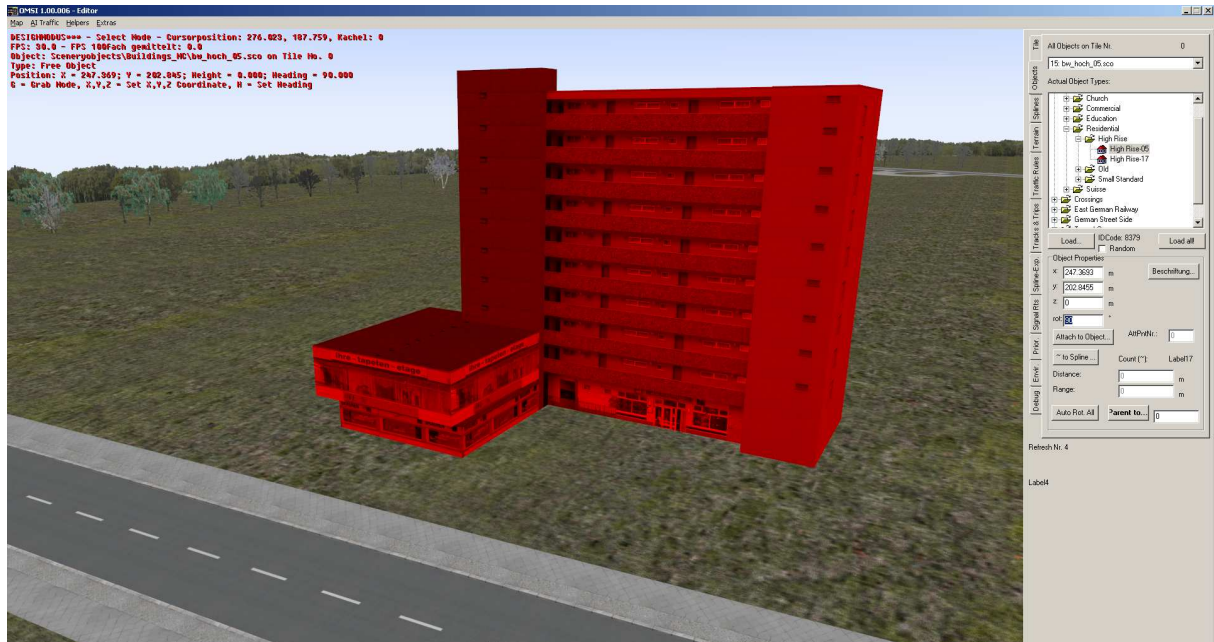
Now select the object type "High Rise-05" as shown in the screenshot. Then press the [N] key ("new") and move your mouse over the 3D view!



You can see the object moving with your cursor! Click the left mouse button or press [Enter] to place the object.

Now the properties at the right corner are filling with figures!

There you may enter e.g. the exact coordinates and rotation with numbers! E.g.: enter 90 in the field "rot:". Now the text will change into blue color to show you that this change was not confirmed yet! Press [Enter] to confirm it. The object rotates 90° around the vertical axis:



The same way you can also change the three coordinates.

Of course you can also move or rotate the object with the mouse: Press [G] to grab or [R] to rotate the object as you like. Then press [Enter]/left mouse button to confirm or [Esc] to undo this change.

Important: In most cases, the z coordinate will be measured relative to ground. But some objects (intersections, bridges) need an absolute height (above zero level of the map).

While the object is marked red, it is selected. If you want to deselect it, just click on an area without objects. If you want to select it again, click onto the object – while you are hovering about an unselected object, it will change into blue color.

While there is an object selected, you can change its type by clicking onto another type in the tree view.

You can delete an object by pressing the [Del] key.

Important: The key [N] works a little bit different, if there already is an object selected! In this case, you will copy the current selected object so the new one will have the same height, rotation and properties!

1.3. Trees

Apart from some exceptions the most trees have special properties, which will now be demonstrated.

Make sure that you have not selected an object and choose the object type „Trees LQ \ Deciduous \ Medium Size \ Tree Medium 01“. Now place this tree object:



The object itself is just a small green box at the tree's base. This you can select, grab, rotate or delete. The tree itself can *not* be selected.

It has the following background: All trees are joined in OMSI internally to increase the performance. A second advantage: You can create trees with different heights and widths:

Click onto "Options...". There you can change the texture, the height and the ratio (width divided through height).

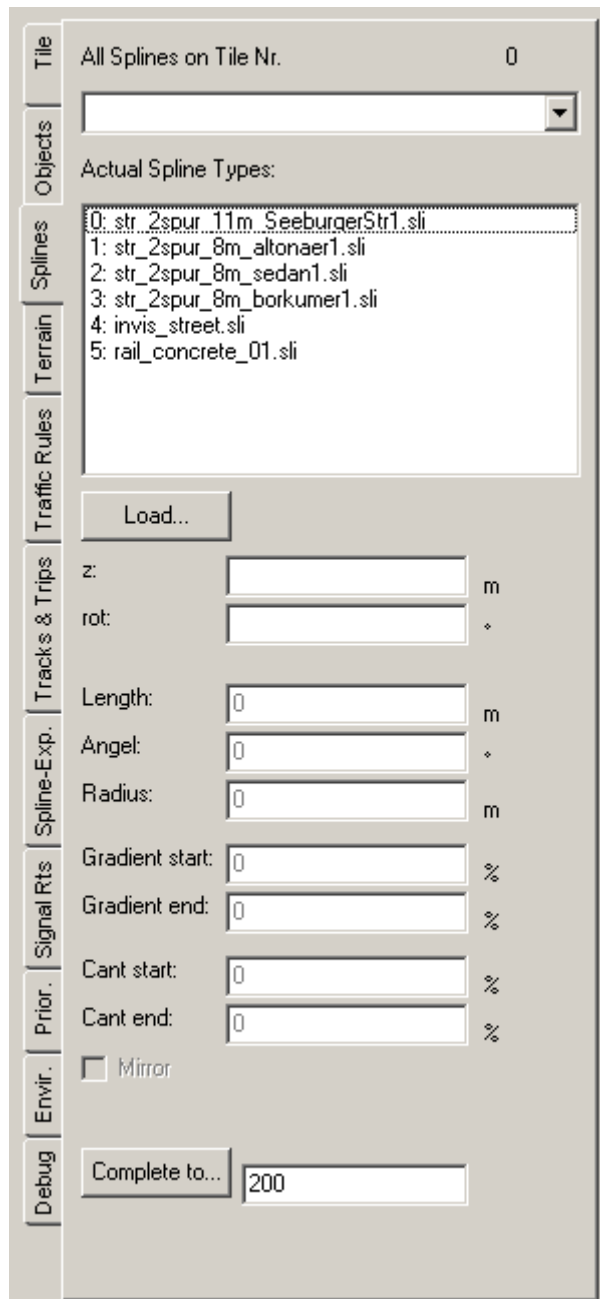
These values will be changed by random if you place a new tree.

Additionally, you can activate the check box "Random": Every new added object will be selected at random from the group of object types – so you can easily create a complete forest with randomized trees only by pressing [N] and placing with left mouse button!



Like I said at the beginning, not all tree types are „real“ tree types – „Chestnut 01“ ... „Chestnut 03“ are only „normal“ scenery objects, but that’s no problem...! ☺ You just cannot change their heights/widths.

1.4. Splines / Streets



In addition to scenery objects, there are splines, too. Each spline consists of a profile defined by the spline type, extruded along a straight or curved line. You can create streets, sidings, walls or railroads with this technique.

Change to the tab "Splines" at the right border. This tab looks similar to the "Objects" tab, but there is no button "Load all".

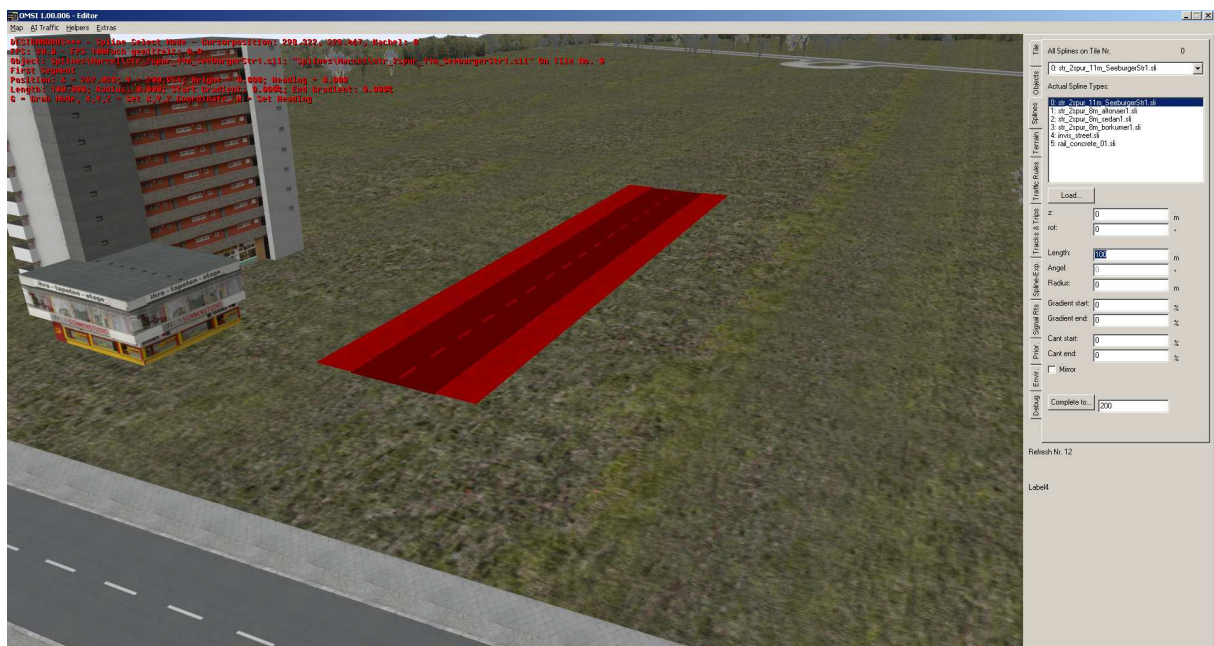
The placing of splines follows the same principles as placing objects.

Select the profile “str_2spur_11m_SeeburgerStr1.sli”, press [N] and place the new spline segment. Click with the left mouse button to confirm your placing.



Rotating, adjusting of height, change of type, deleting and entering of properties will be done in the same way as with scenery objects.

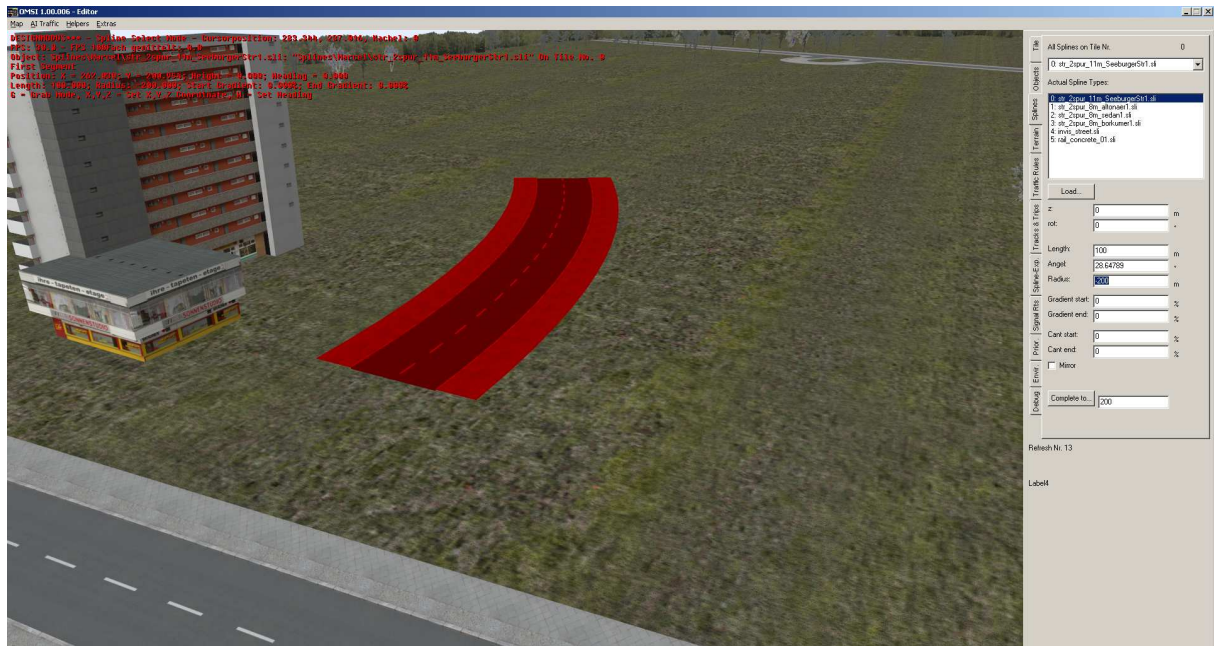
Of course you can additionally shape the splines: First type “100” in the field “Length:” to adjust the spline’s length to 100m and press [Enter]. You may also use decimal point values, e.g. “10.5” for ten and a half meters of street.



Important: The maximum length of spline segments should not exceed 300m! Otherwise you can get graphical problems (e.g. invisibility in special perspectives, incorrect lighting or collision detection).

Of course you can also bend the street: Enter a radius unequal to 0! If you enter a positive radius, you will get right handed curves, if it is negative, the curve will be left handed.

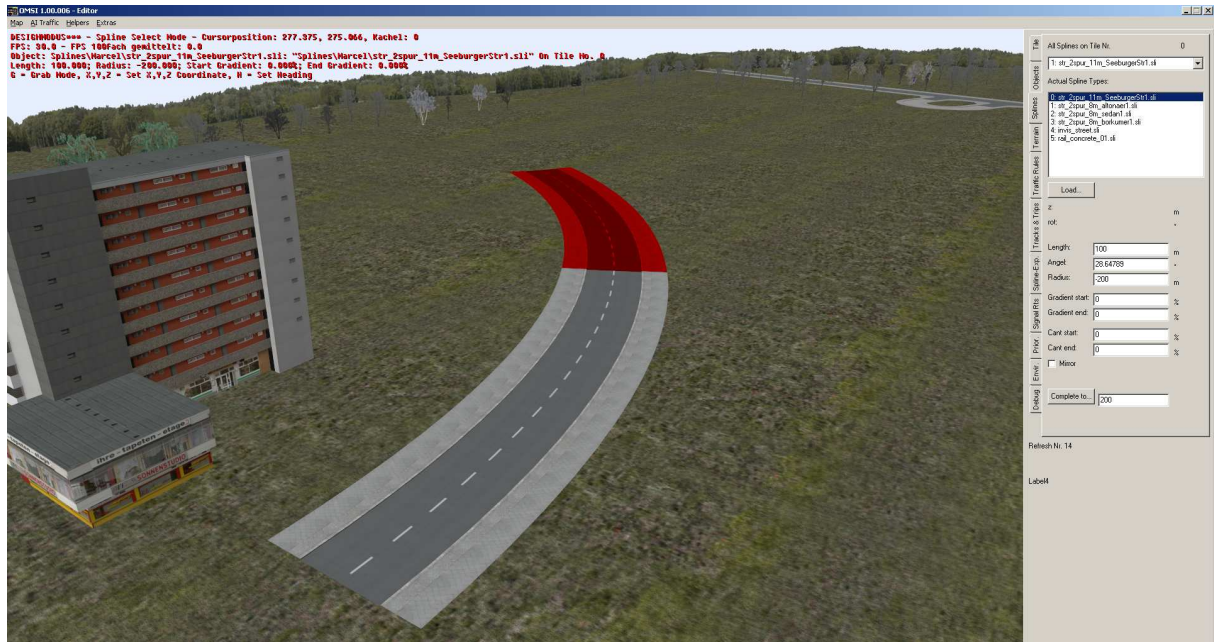
E.g.: Enter "-200" and you will get a left handed curve with 200m radius:



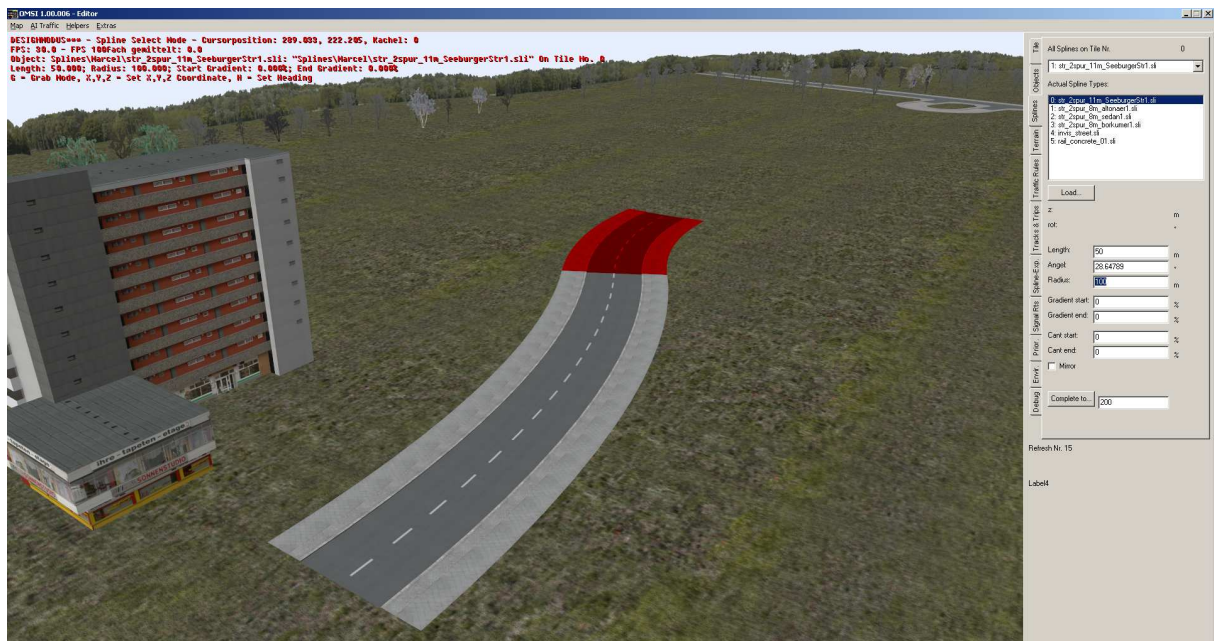
You can enter a gradient or cant in the same way. In this case, you will have to set values for each end: One defines the gradient/cant at the beginning of the spline, the second one will define it at the end.

Of course, you can also add further segments:

While having selected the last segment (redcoloured), press the key [N]:



A new segment will be added, which has the same properties as the last segment. Now you can adjust the properties, e.g. length = 50m, radius = 100m:



Important: The first segment of this chain is now the „master“ and all following are the „slaves“. The starting height and direction can only be adjusted for the master segment, all slaves are just following. If you change an “earlier” segment, the “later” segments will follow these changes.

Important: If you would like to start a new chain of spline segments, you have to press [N] while *no* existing segment is selected!

1.5. Intersections and streets

Now we will link streets with intersections!

Intersections are special scenery objects with so-called "paths". Paths are straight or curved lines on scenery objects (e.g. intersections) or splines (e.g. streets) with a certain width to define traffic ways.

There are different types of paths, e.g. street or pavement. A path can be usable for one or two directions, it can be fitted with traffic rules (speed limit, traffic priority or blinker), can have a special traffic density and it can be used for precise placing of intersections and streets, because they define the linking points.

Change to the tab "Traffic Rules" for a short look on the traffic paths:



You can see the pavement paths in green and the street paths in brown. In the map editor, each path has only one meter width, but the real width usually differs.

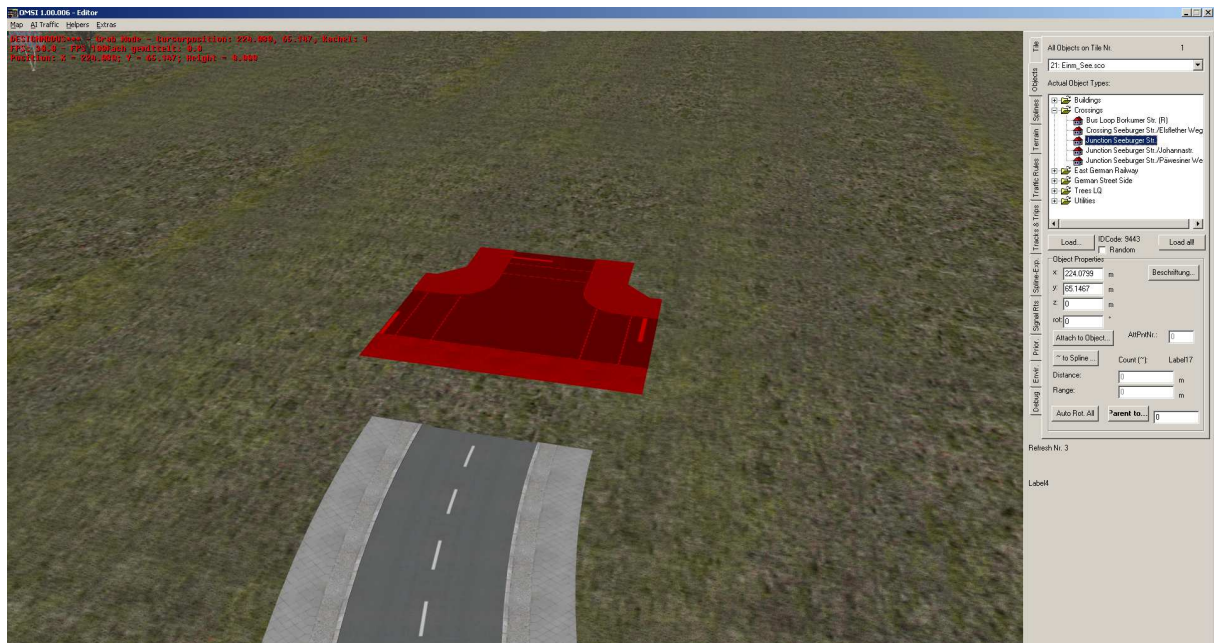
You can also see the direction of the paths: The street paths are "one way paths", the pavement paths are "two way paths".

Now change to the "Object" mode and add an intersection.:

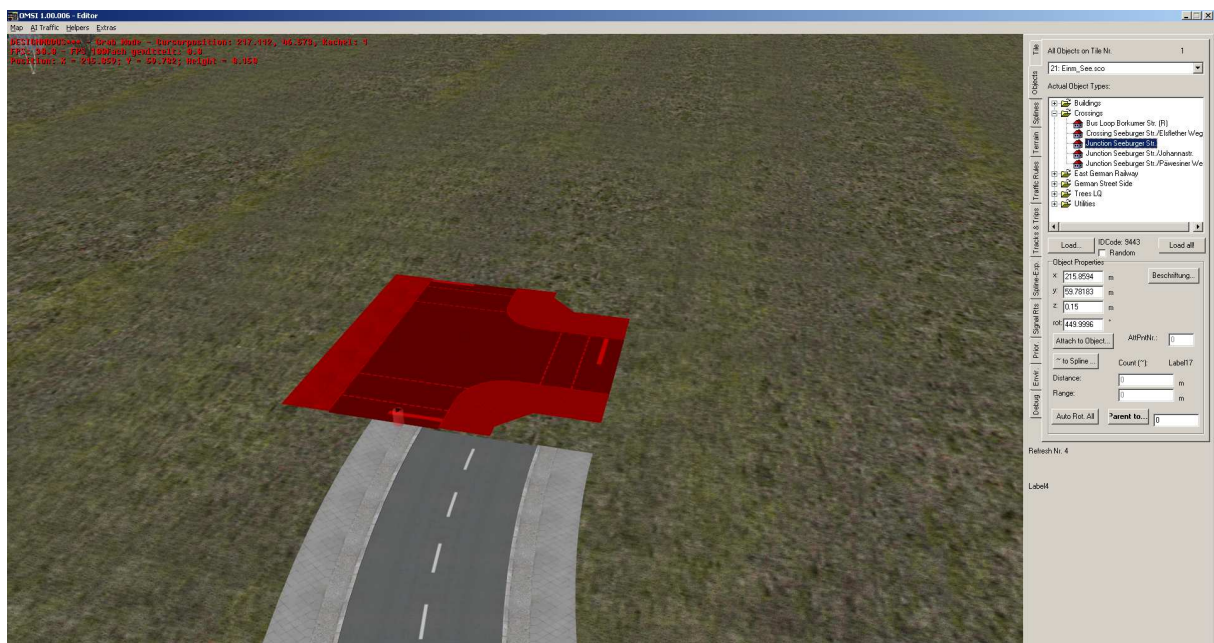
Change the object type to "Junction Seeburger Str." which fits to the already added spline type.

With the current version of OMSI Editor, it is not easy to find fitting intersection/spline pairs, you have to experiment a little bit. In each case you should check the fitting in the "Traffic Rules" mode! On the map "Grundorf" you will see some standard streets and intersections which are compatible to each other. In the appendix you will find several fitting spline and intersection types.

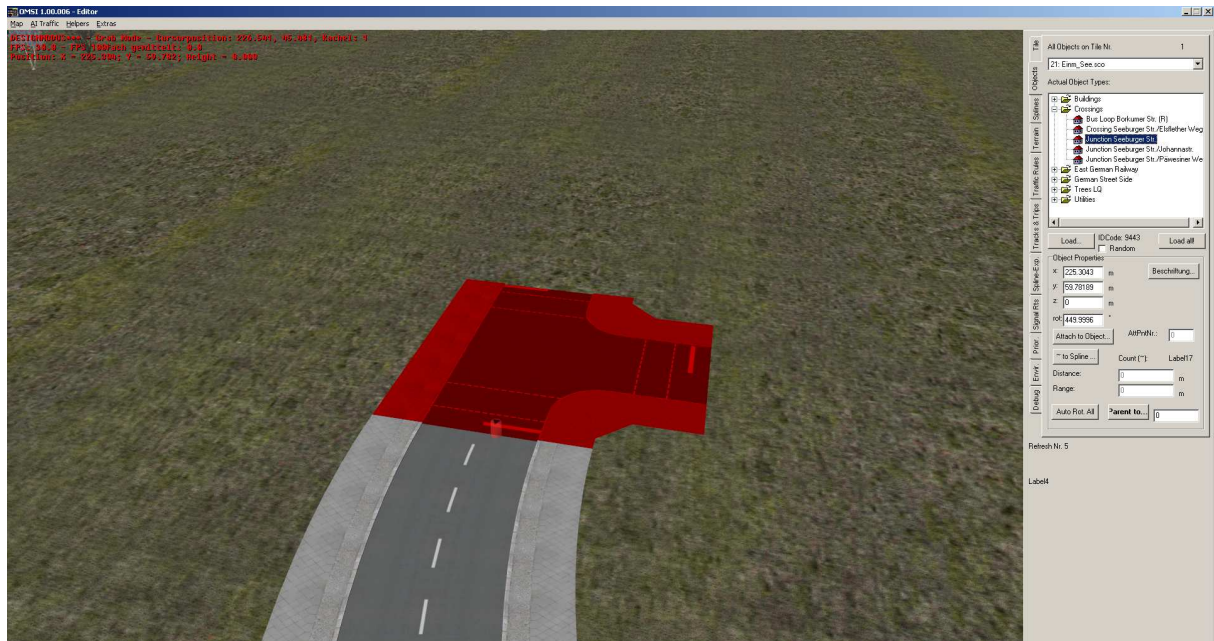
Press [N], to have an intersection "at hand":



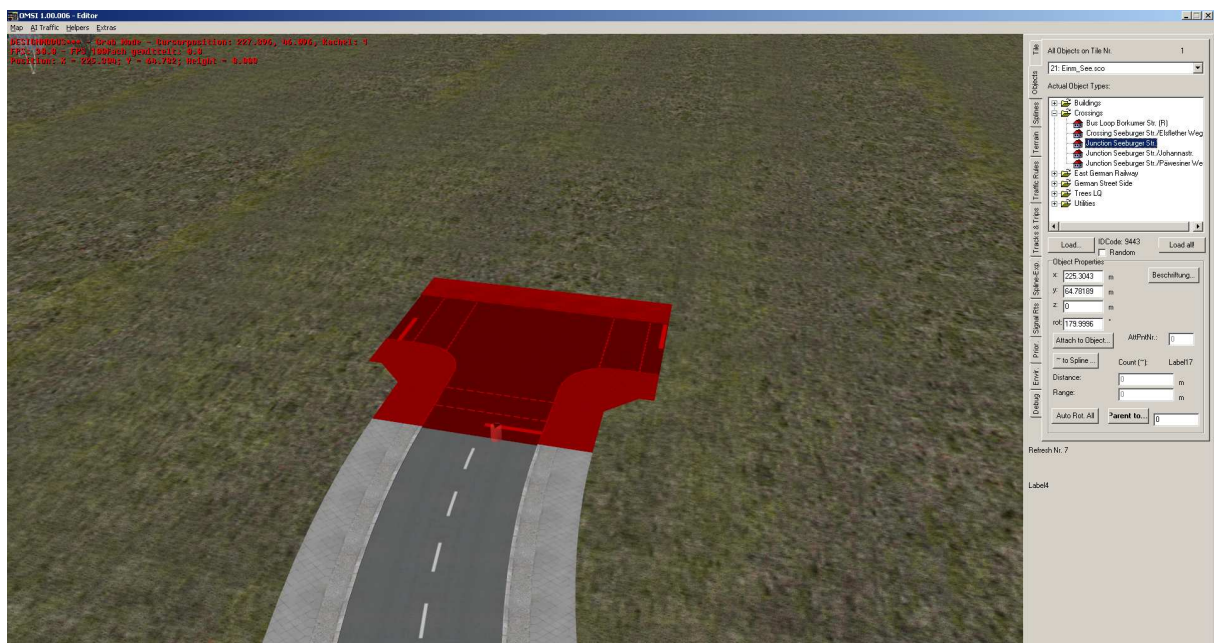
Now move the mouse to the street end! If you are near enough, the intersection will link automatically:



But this position does not make sense, but you just have to move the intersection a little bit more to the right, then it looks quite better:

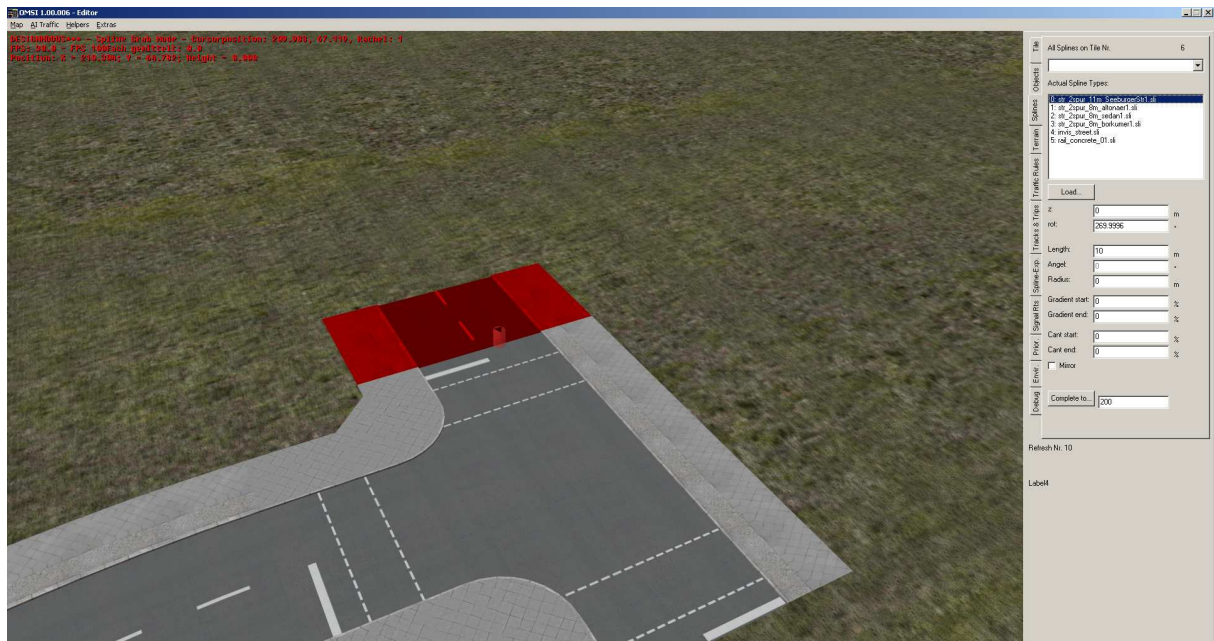


But what can you do, if the intersection has to be positioned "lateral"? It is very easy: Just press [F9] until it will be positioned correct (while still having it "at hand")!

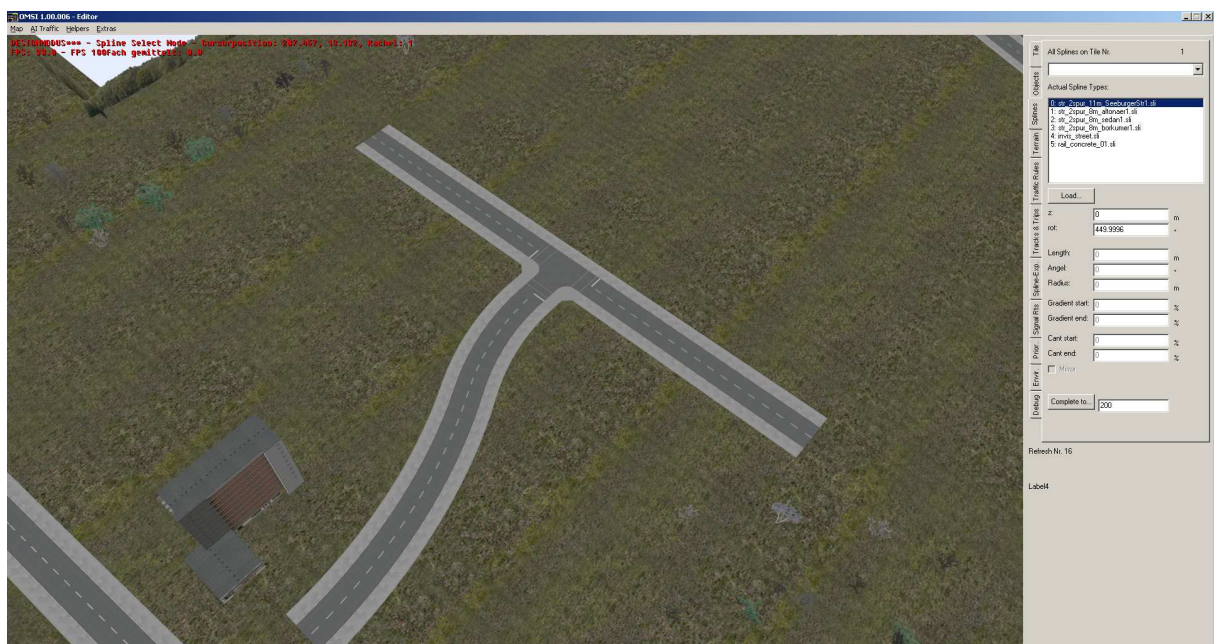


After confirming with the left mouse button, the intersection is positioned perfectly fitted in the scenery!

Now change to the spline mode and create a new spline segment. Linking this to the intersection can be done the same way:



Now you can add two streets with 100m at both ends of the intersection:

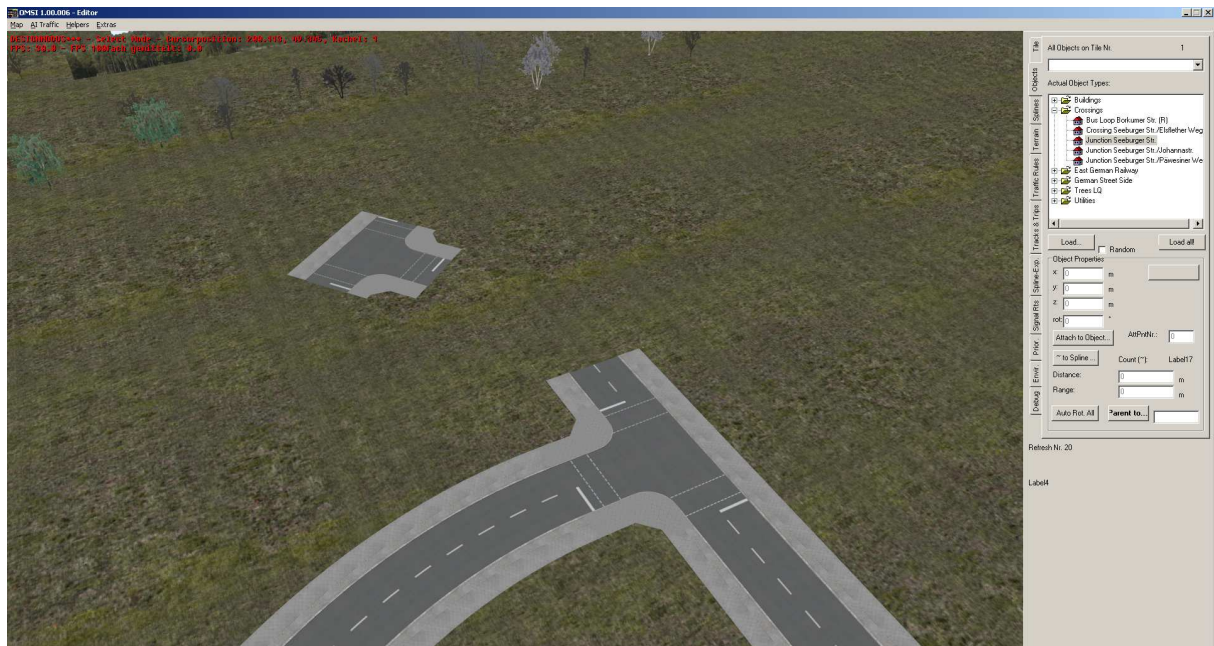


Some stations, buildings, some trees and the new OMSI map is finished already! ;-)

OK, not really... but I think, you now have learned how fast and easy you can build up a new network of streets and intersections!

And we will continue:

Please reduce the length of the left straight street back to 10m and place a second intersection like shown in the screenshot:

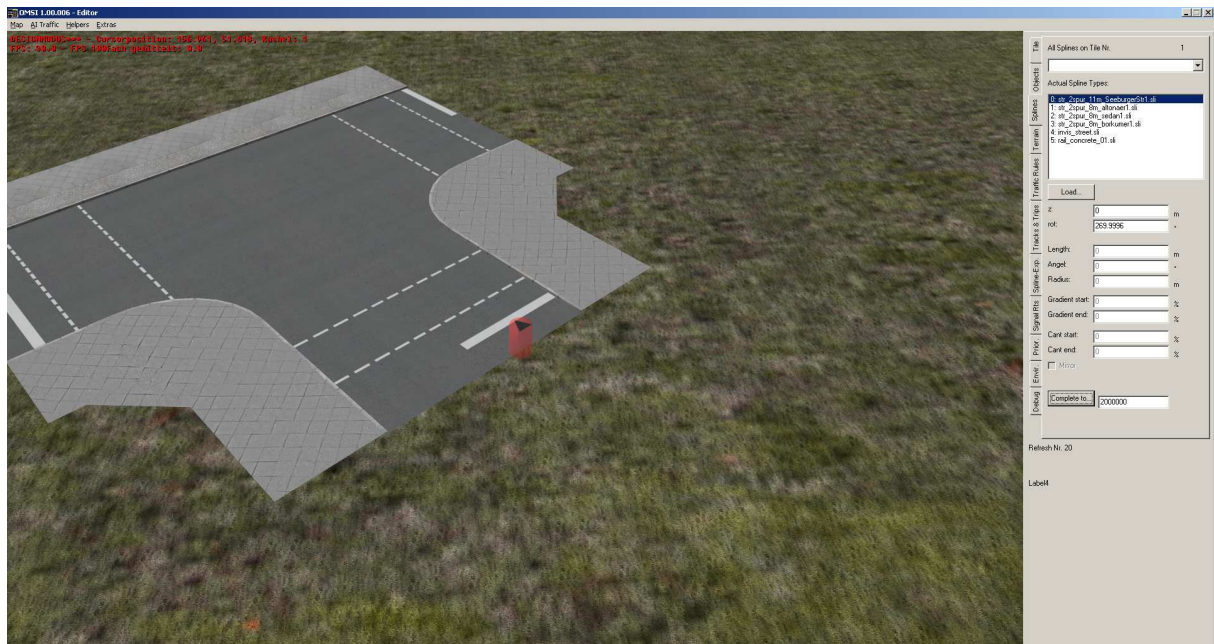


Assuming that this intersection is already placed and cannot be moved.
How can we now add fitting streets in between?

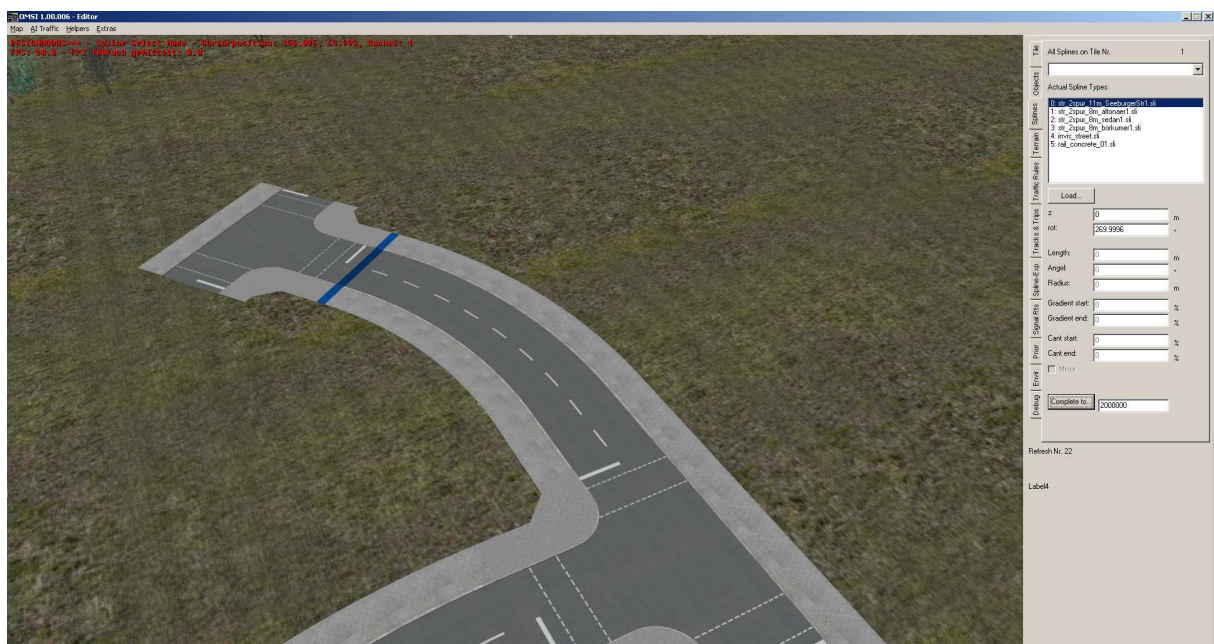
This is also not too difficult (the only restriction in the current version: both ends have to have the same height): For these purposes, you have the function "Complete to..." on the spline tab. Enter a huge radius 2000000 into the field next to this button, because we just do not want any radius limit. Now click onto "Complete to...", then on one path end...



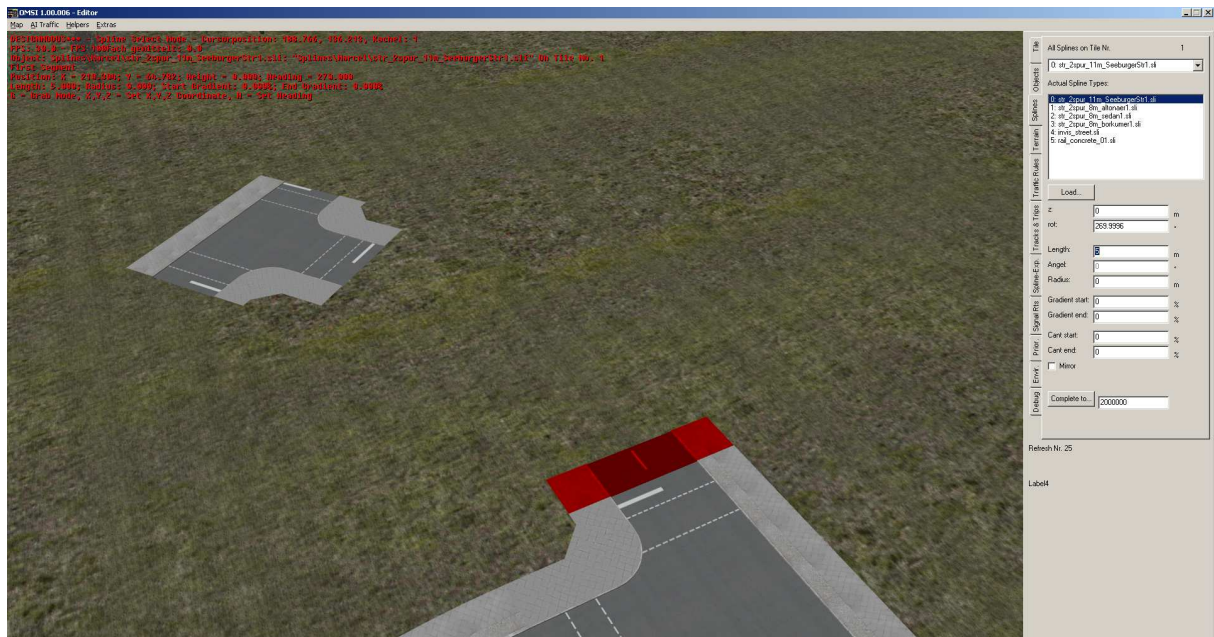
... and then on the adequate one on the other end (same path, else it will not fit!):



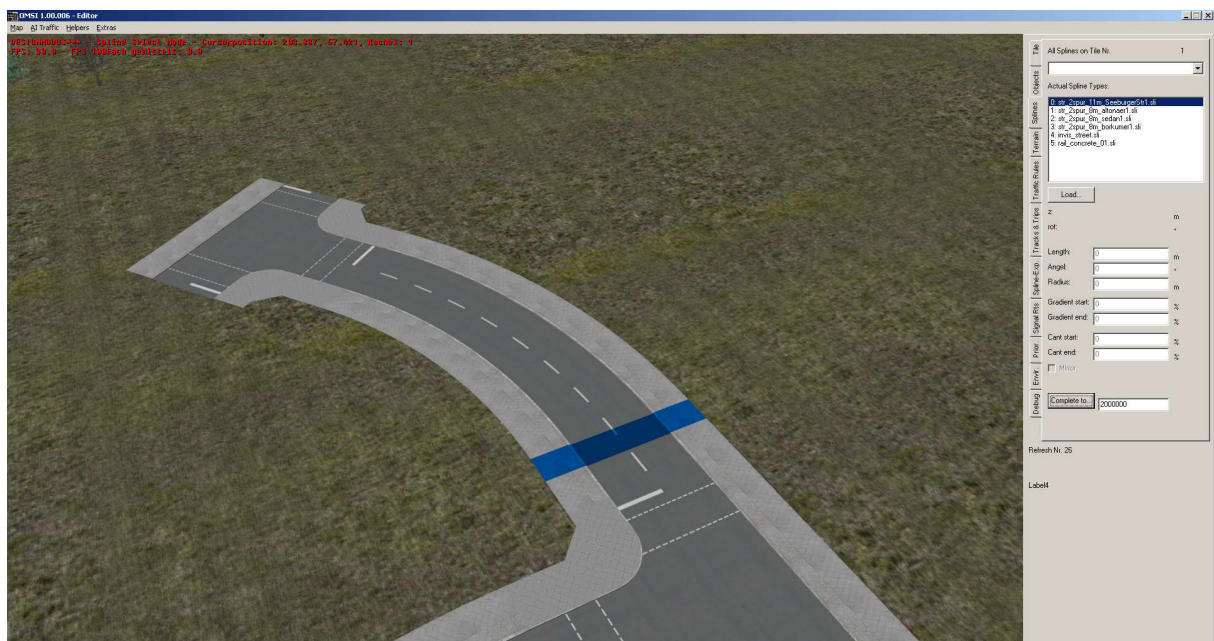
Instantaneous, the missing segments will be added!



Important: In this case you can see a very short spline segment. This *may* cause some troubles with the AI traffic. Here you can solve this if you delete the added segments and shorten the first segments to 5m:



Now the street looks quite better:



Important: In the current version of the Map Editor, the current tile (on which the centre point of the map cam is located) has to be the same where the link point is, too. So in some cases you first have to center the map view on the link point before you can click on the link point.

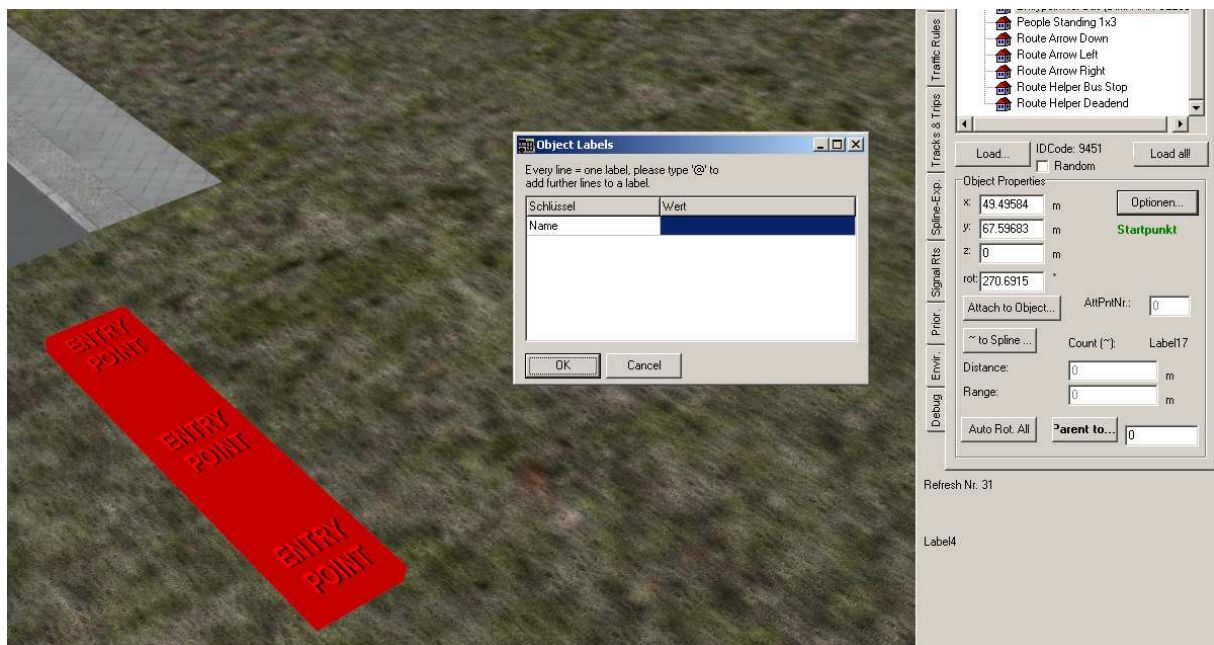
The result looks quite good, doesn't it?



Choose the object type “Utilities\Entrypoint for Bus” and place it near the street (not *on* the street, otherwise you can get collisions with AI cars!)

This is an entry point, which can be used to place the bus!

You have to give it a name! Select it and click “Options...”



Enter a name in the blue marked field!

Important: You may give the same name to several entry points! In this case, OMSI has some more possibilities to place your bus if some of these

positions are occupied by other buses. You can see this situation e.g. at the terminus Nordspitze Bauernhof:



Save it, close OMSI Editor and start OMSI (with the map "My Map") to do a first test run on your new streets! You will see that OMSI automatically generates AI traffic on the new street and intersection paths!

Important: Until you have visited the map for the first time, it won't be listed while you select "Load last situation on map". In this case, you should select "Load map without buses"!

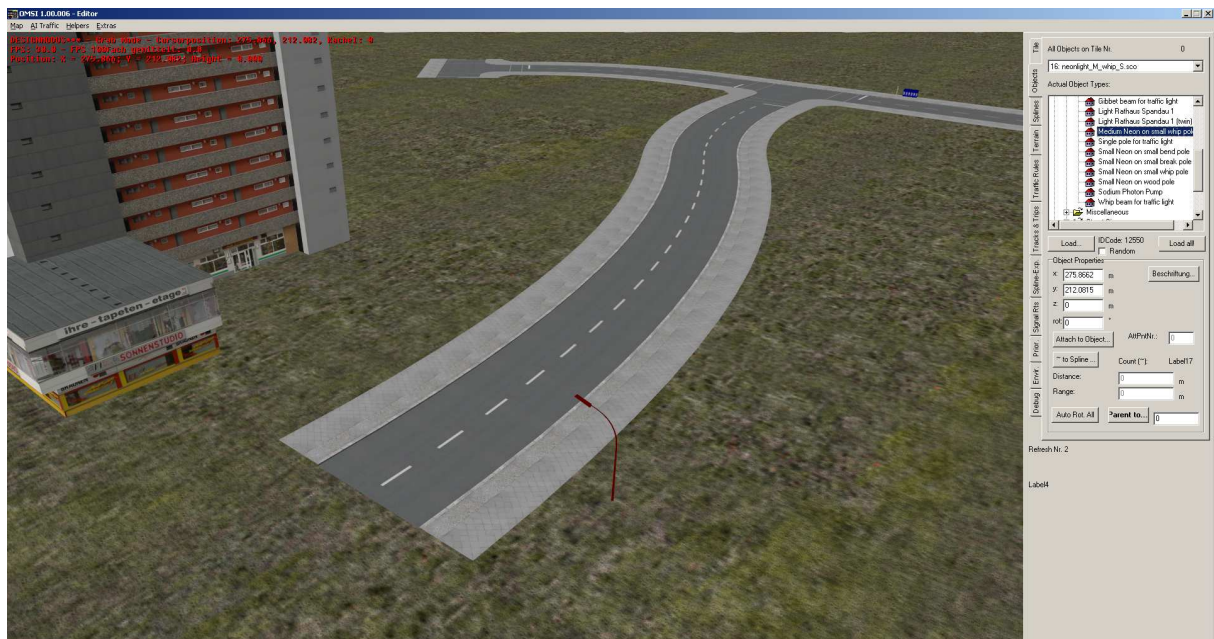


Lesson 2: Enhanced Object Editing

2.1. Attach Objects At Splines

For an easy placing of e.g. street lights, there exists a special function which allows you attaching a row of objects along splines. We will now use this function.

Place an object of type „German Street Side \ Lights \ Medium Neon on small whip pole“ near the first built street and confirm:



Now click on “~ to Spline...” and then on the *first* spline segment. Now click on the beginning of the street. With [G] you can correct the position.

Important: Unfortunately, the Map Editor runs not completely stable and still has some bugs. E.g. often the object is invisible directly after attaching – just confirm it without seeing it and correct the position with [G] after that. Also it can happen that the position of the object is totally wrong or the map editor crashes, so please save the map before do such actions. You should also attach it to the master segment, if possible. In the worst case, you have to abandon this function.



At next, you have to adjust the height.

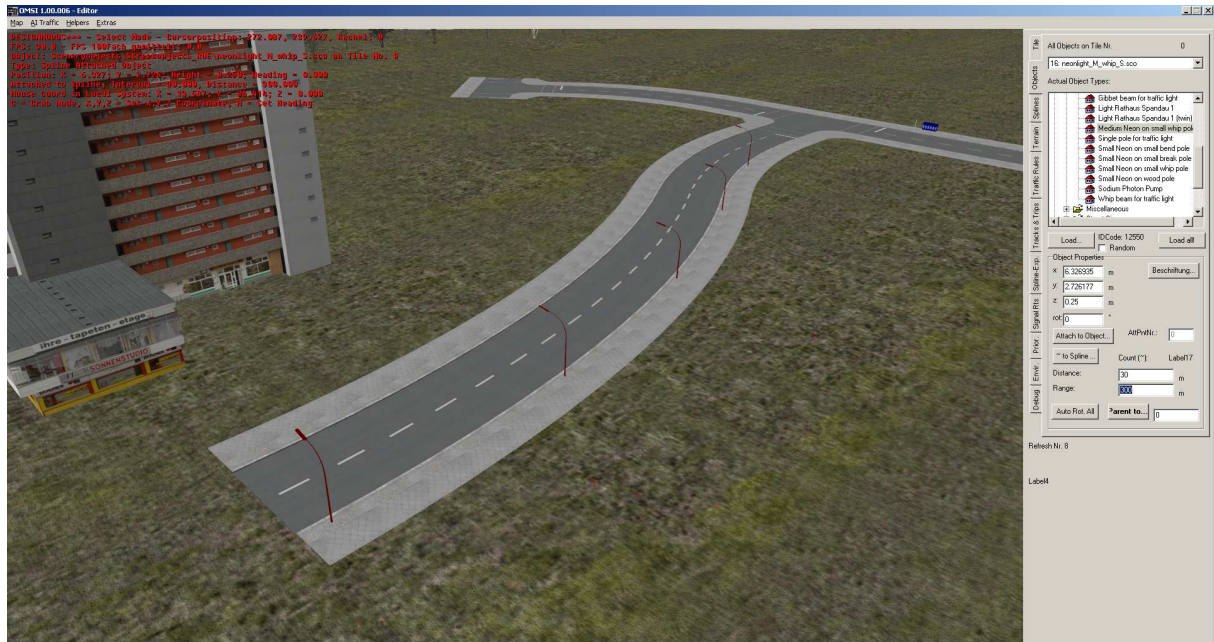
Important: All pavements have a height of 0.25m! (Only a few very old and unused objects/splines have 0.3m.)

Now type in the height „0.25“ and confirm with [Enter].

Important: The spline attached objects will be rotated with the spline! That means that usually you only need rotation angles of 0, 90, 180 and 270 degrees.

Now we want to place a complete row of lights along the street!

Enter “30” next to “Distance:” (every 30 meters one light) and “300” next to “Range:” (the street is obviously shorter than 300m, so the lights will be placed just until the end of the road). Confirm with [Enter]:



Now all instances form one unit and can be edited together like one object. If you want to move them, please mind that you control the first object of the row! Rotating with the mouse is not possible yet, but like you can see above, usually only "even" values are necessary here which can be edited better with keyboard.

In the same way, you now can add lights to the other streets.

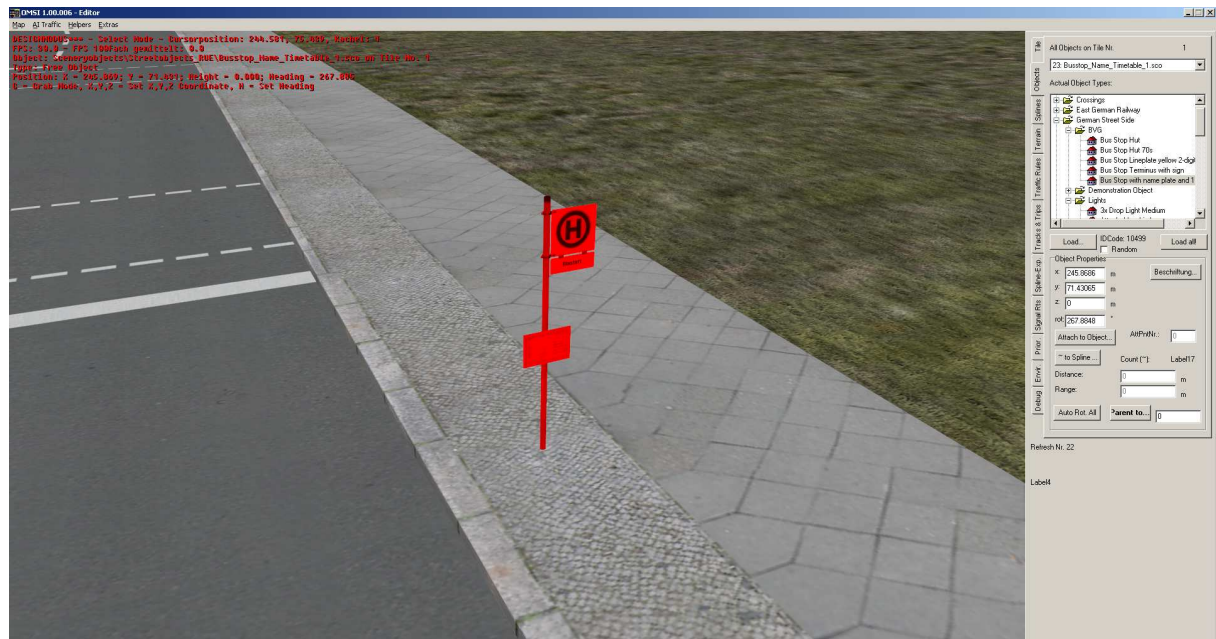
2.2. Attaching Objects At Objects

Similar to the previous function, it is also possible to attach objects to other objects, if they are fitted with so-called attach points.

Important: For this purpose, the order of creation must fit! Always create the "parent" object *first*, *then* the "child" object!

We will now demonstrate this function with a bus stop. Please mind: The bus stop we will place now is only a "normal" scenery objects without interactive functions! Later, you will also learn how to place an interactive bus stop!

First of all, place a bus stop flag with time table and name plate (object type: „German Street Side \ BVG \ Bus Stop with name plate and 1 timetable“):



Now we want to place a line card on top of it!

Deselect the bus stop flag (important! Else, you will change the object type of the already placed flag!), choose the type "German Street Side \ BVG \ Bus Stop Lineplate yellow 2-digit" and press [N] (don't confirm yet!). Now, you have the line plate "at hand", perhaps it is too small to see it. *Without confirming*, click on "Attach to Object..." and then on the already placed bus stop flag:



Ok, that's not the best place for the line card! ☺ But you have some more possibilities/attach points; this one sometimes is used by the "bridge" if

there are more than four line cards necessary. So enter now "2" as "AttachPntNr.:"!



Now the line card has moved to the right place.

Important: Unfortunately, the map editor has also a bug in this function: If you move the "parent" object, the "children" will not move instantaneously with their parent but only if you reload the map or if you select the child. Press [G] (making not much sense hence this object is attached) and [Enter] – this action will refresh the child's position.

By the way: Child objects can also be parent objects for further child objects! Watch the bus stops at Moritzstraße, you will see that some line cards are attached to one of these line card bridges, which is attached to another card bridge which is attached to the bus stop flag. Some of the bus stop flags are also attached to a street light pole.

Furthermore, you can also rotate the child object (e.g. these street light pole attached bus stop flags). In this case, "0" means, that the child is placed as designed.

2.3. Label Objects

Now label line card and bus stop sign as you like! Select it, click onto "Label..." and enter your favorite text!

Please mind: An "@" means "line break". Furthermore, you can enter (only on bus stops) a white "U" on a blue box (underground metro), a white "S" on a green circle (city railway), or a white "X" on a purple circle (express

buses) using the symbols {, [and]. E.g. for the following screenshot I entered: {Walter-@Vogel-Platz.



2.4. Place Traffic Lights

In OMSI Traffic lights consist of two parts: The traffic light control, which has to be included in the intersection scenery object, and the traffic light units for themselves, which you can see over the intersection.

Because the traffic light control is included in the used intersections already, you just have to add and connect the traffic light objects to fit the intersection with a full functioning traffic light system.

First of all, place an object of type "German Street Side \ Lights \ Whip beam for traffic light" in each corner. You find the traffic light poles under "Lights" because of the integration in the Berlin street lights. Adjust the height to 0.25m.



Place these three light beams in the middle of the pedestrian crossings because later we will also add pedestrian traffic lights.

Now place „Single pole for traffic light“ objects vis-à-vis, also with 0.25m height:



So the mountings for the traffic lights are finished. Now we place the traffic lights themselves. First we attach overhead traffic lights direct to the whip poles.

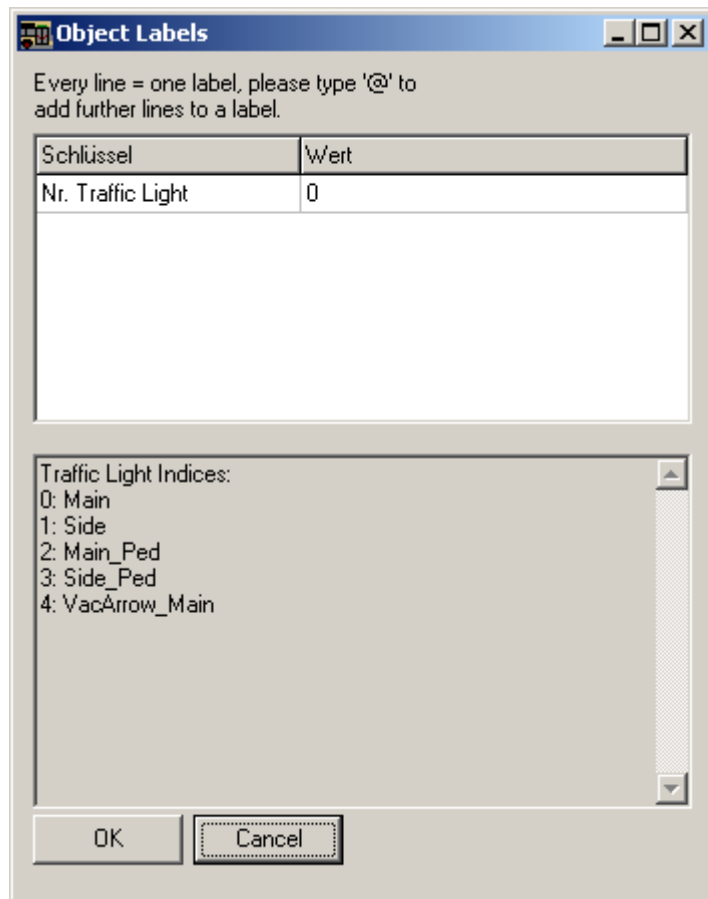
Add an object of type „German Street Side \ Traffic Lights \ Traffic Light Cars Overhead w. Blind” and attach it at the whip pole with the attach function as shown in the screenshot:



You can see that the traffic light does not operate yet, because it does not “know” its intersection! Click on “Parent to...” and then on the intersection. Next to the button “Parent to...” you can now read the IDCode of the intersection. This is a unique number for each object or spline for identification. You can read the object’s IDCode beneath the type selection box!

If there will be any refresh (e.g. if you add further traffic lights or move anything), the traffic light will now start working.

Now you have to check if your traffic light was allocated to the right traffic flow. For this purpose, click on “Options...”:



You can see the list of possible traffic flows in the grey box: Main, Side, Main pedestrians, Side pedestrians and the green vacation arrow for cars turning left from the main direction (VacArrow_Main).

Important: If you can see *no* list, you either have not parented the traffic light to a intersection or the intersection does not support traffic lights!

As you can see, the "0" is correct: The traffic light controls the main direction.

Now press [N] *while the last created traffic light is selected*, so the new overhead traffic light will have the same properties as the last created one (including "parent"). Attach it to the opposite whip pole and check, if it also has a "0" as "Nr. Traffic Light".

Now add a third traffic light the same way and attach it to the whip of the side lane!



Give *this* traffic light a "1" as "Nr. Traffic Light" hence it shall control the side lane!

Now watch the traffic lights in the editor and then in OMSI: They should control the traffic properly!

Important: The traffic light system of the intersection will become active if you parent at least one traffic light to the intersection! Otherwise, the traffic behaves like if the traffic light does not exist (accordingly to the given priority rules).

Now we want to add the missing traffic lights.

All other traffic lights have „normal height“ and will not be attached to the poles but moved freely to the proper position. This will be done for a higher flexibility.

Create a new traffic light using one of the existing traffic lights (which have the parent property set already) and place it near to a whip pole. Select the type "Traffic Light Cars":



Now enter the standard height for all traffic lights – “2.8” – and turn it that it looks into the right direction:



Now move it exactly to the pole. You should use a perspective similarly to the next screenshot:

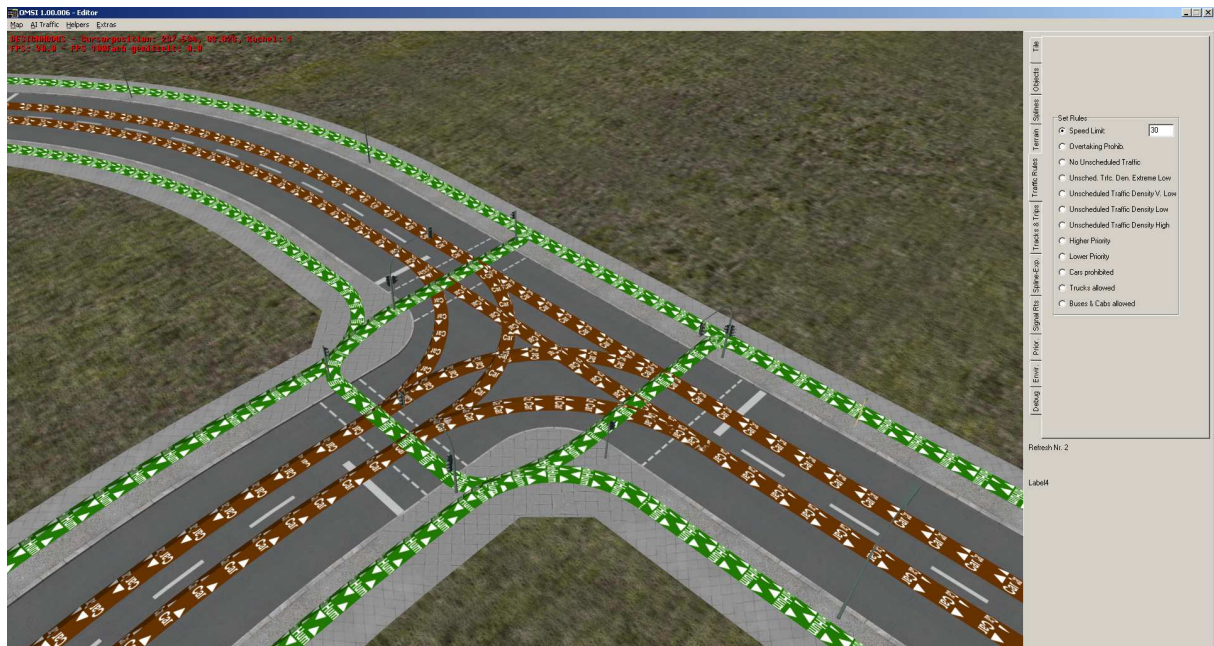


You will get the best result if the both mountings “touch” a little bit into the pole.

The same way you can also place the two other traffic lights. Please mind, that you have to enter a “1” again for the traffic light for the side lane.

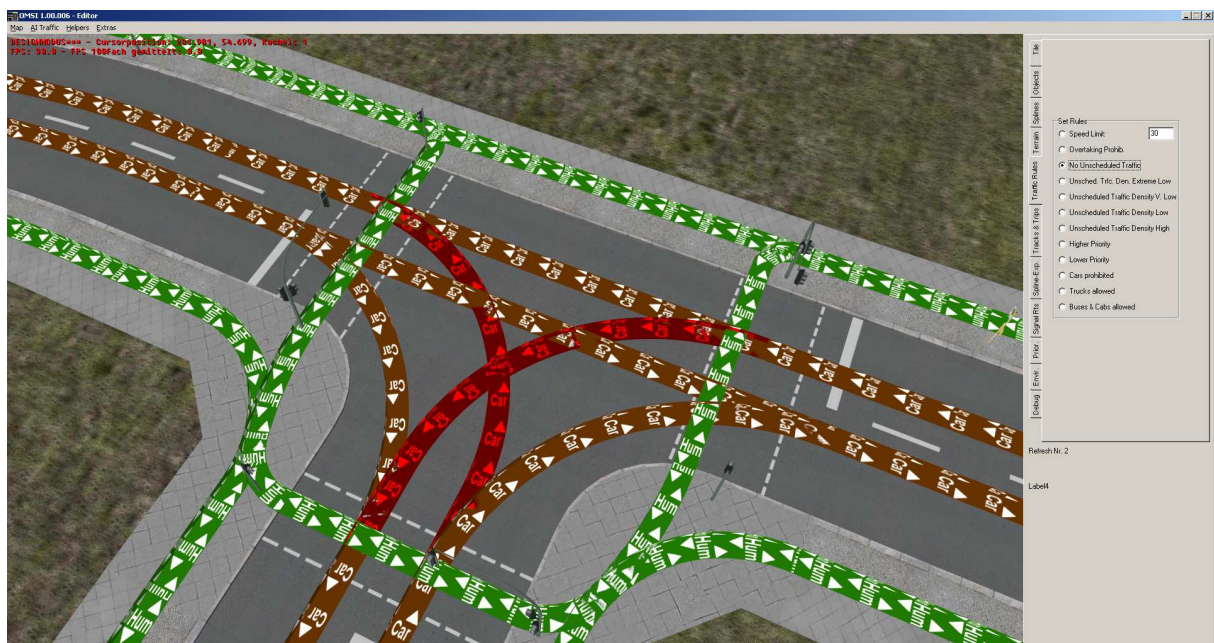


Last but not least, you should also install pedestrian traffic lights. These will be placed as you can see in the next screenshot and will get a “2” for the crossing parallel to the main route and a “3” for both crossings parallel to the side route. The height will be 2.8m, too:



At the right hand side, you can see a list of different traffic rules.

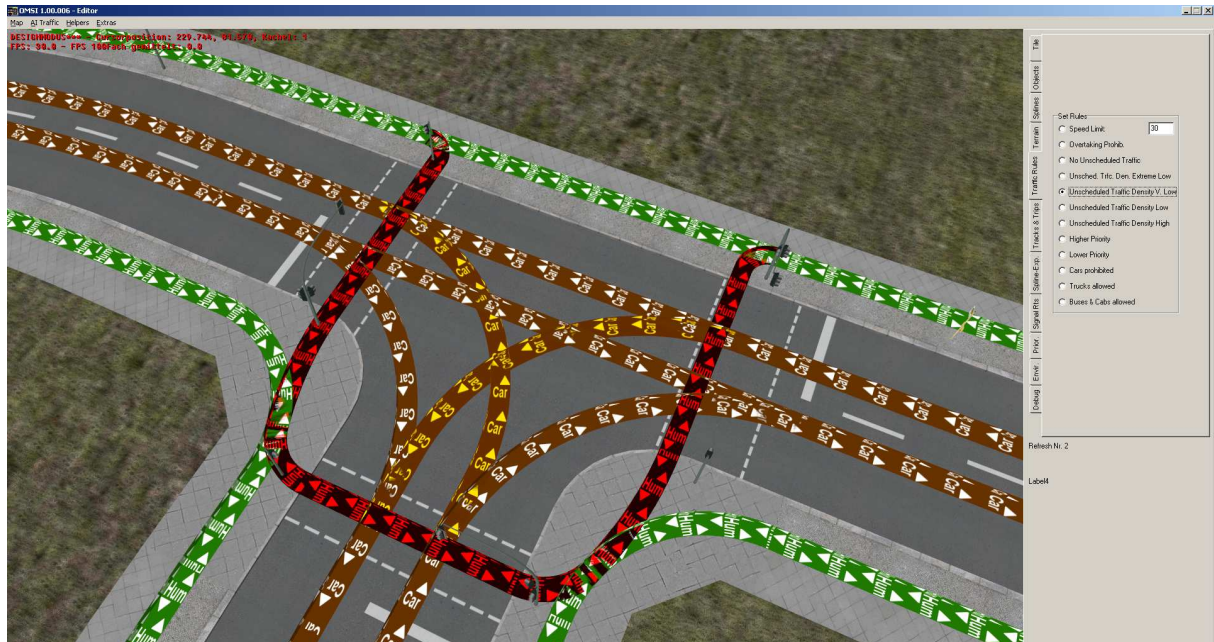
First we would like to prohibit left turns for the unscheduled traffic. (Please mind: all these prohibitions are only valid for unscheduled traffic – scheduled traffic will always use “its own” paths!) Choose now “No Unscheduled Traffic” and click or “paint” over the affected left turn paths. Pay attention that you also mark the short straight path at the side! If you want to remove markings, hold [shift] while painting!



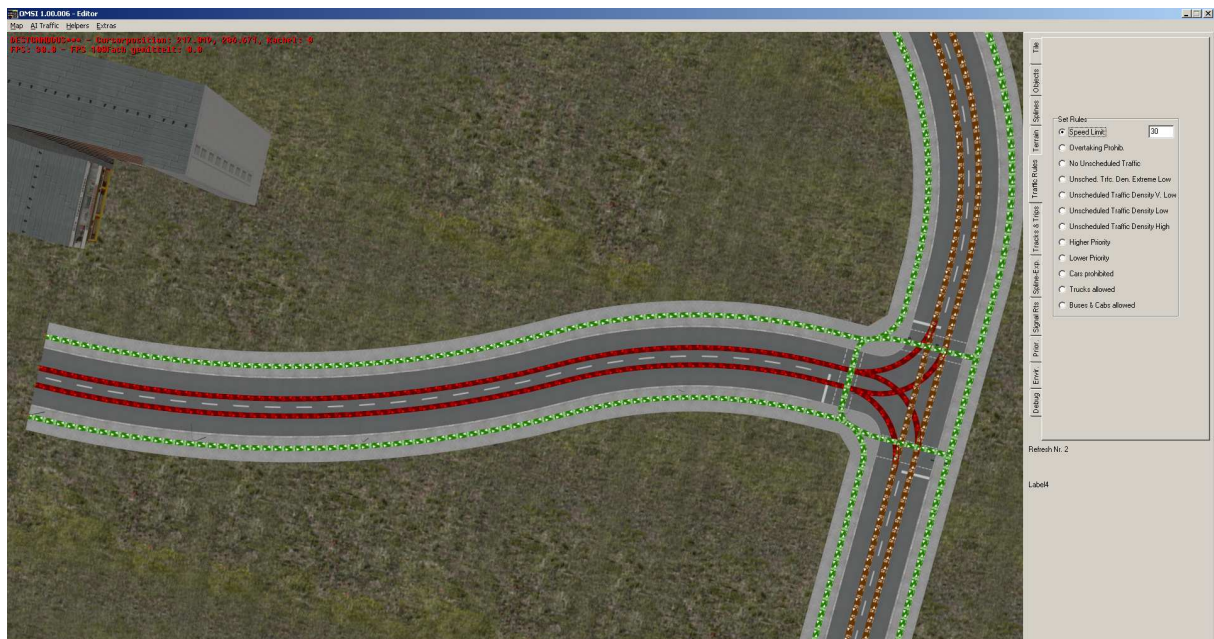
Now we want to reduce the pedestrian density on the pedestrian crossings.

Choose "Unscheduled Traffic Density V. Low". The paths you have marked already change into yellow. This means that they are already marked with a special traffic density, but this is not equal to the selected one!

Now paint the pedestrian paths on the intersection and pay attention on marking from junction to junction like in the screenshot:



This principle of the red and yellow markings can be also demonstrated very well with the speed limit. We will now change the speed limit on the side lanes to 30 km/h: Choose "Speed Limit" and enter "30". Now mark the paths of the side lanes:



Assuming that you now would like to change the speed limit of the main road to 60 km/h: Enter 60 now. If you have done that, all already marked paths are yellow now. It is the same meaning: They also have a changed speed limit – but it is not the same as the entered one.

Now the meaning of all traffic rules:

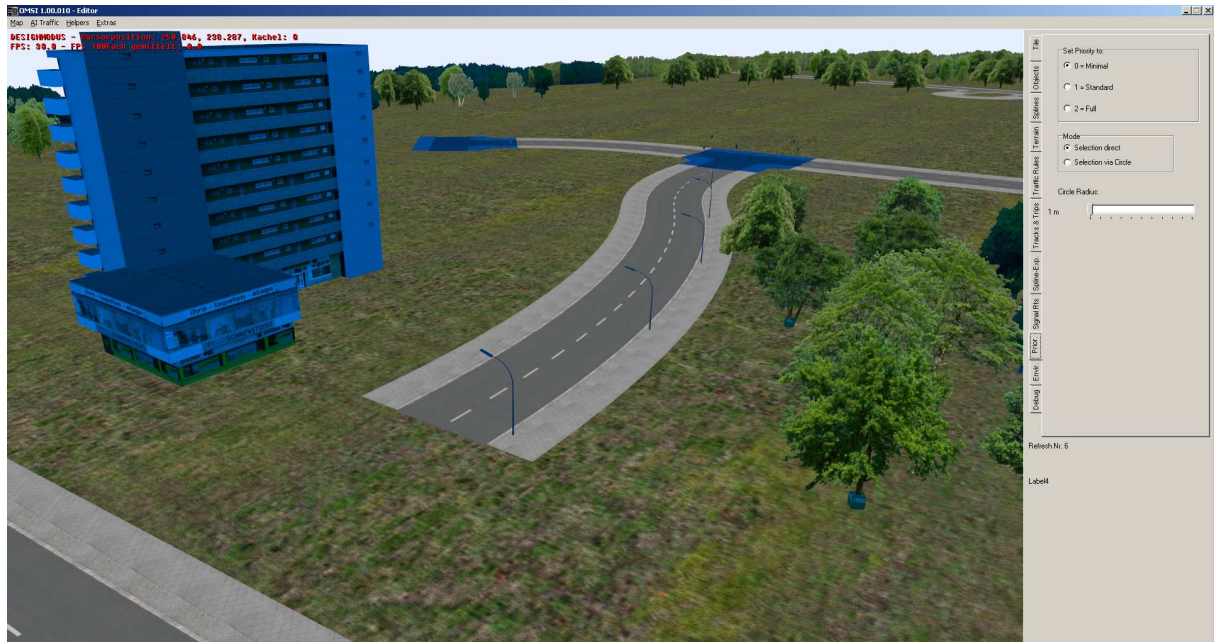
- **Speed Limit:** Change the speed limit. The standard speed limit on streets is 50 km/h.
- **Overtaking Prohib:** Do not use it. AI traffic cannot overtake yet.
- **No Unscheduled Traffic:** On the marked paths unscheduled traffic is not allowed.
- **Unscheduled Traffic Density Extreme Low/Very Low/Low/High:** Changes the traffic density. You can influence the placing density at the end points, along the whole paths on approaching a new tile and also the decision of the unscheduled cars which junction they should use.
- **Higher/Lower Priority:** Set the traffic priority higher or lower the standard level.
- **Cars prohibited:** Prohibits normal cars.
- **Trucks allowed:** Allows trucks on that path.
- **Buses & Cabs allowed:** On these paths are buses or cabs allowed.

So if you would like to have a bus lane, then mark “Cars prohibited” and “Buses & Cabs allowed”, if you would like to have a truck lane, then mark “Cars prohibited” and “Trucks allowed”, if you would like a mixed track including trucks, then just add the rule “Trucks allowed”.

2.6. Map Priorities

At the end of this lesson we will demonstrate how to add map priorities to the scenery objects.

Open the tab “Prior.”:



As you can see, all scenery objects are coloured blue.

So you can change the priority selection to 1 or 2 and “paint” in mode “Selection direct” (more precise) or “Selection via Circle” (to select bigger areas). In this example, the house got priority 1 and the whip pole light got priority 2:



Lesson 3: Create New Map And Terrain Editing

After the past lessons on the copy of the Grundorf map, we now want to create a complete new map.

3.1. Create New Map

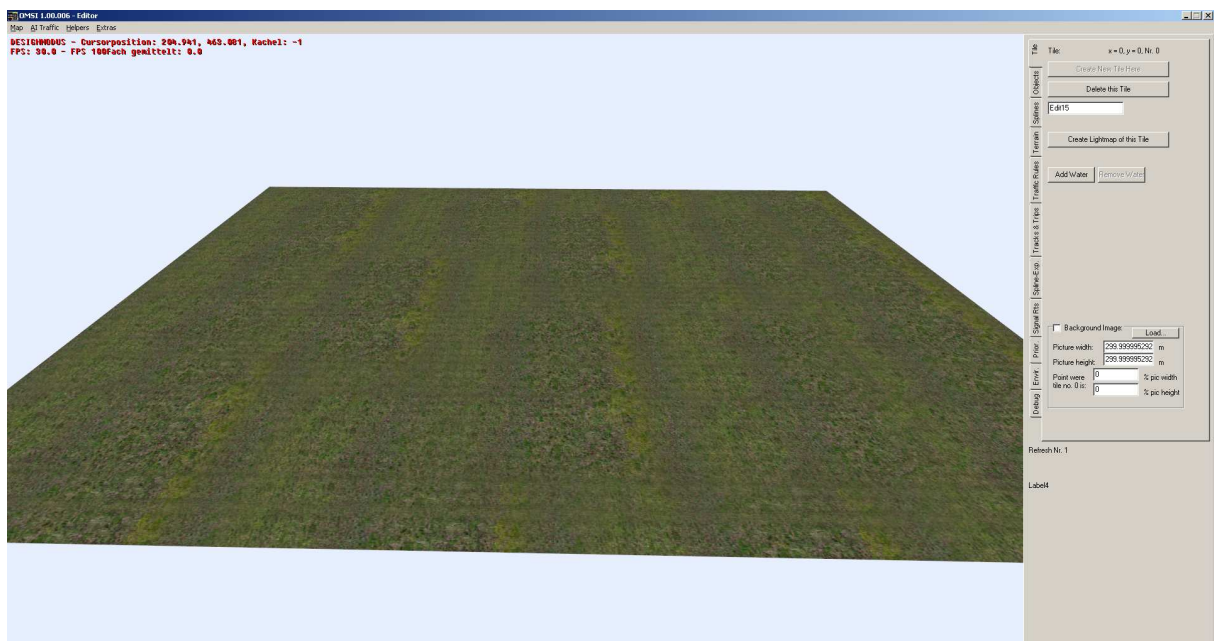
If you want to create a new map, you have to do the same steps as if you copy a map. The only difference is, that the "original" map you will copy, will be the "NewMap" in the Template directory. So the following steps are described a bit shorter.

Open the OMSI directory and change into the subdirectory „template“. You will find there the subdirectory "NewMap". Mark this and copy it ([Ctrl] + [C]). Now go back into the OMSI directory and then into "maps". (You should now see also so subdirectories "Berlin-Spandau_89", "Grundorf" and "My_Map". Now paste the previous copied subdirectory "NewMap" here ([Ctrl] + [V]). You now should see the directory "NewMap" next to the other map directories I listed above.

Now rename the directory into a name of your choice, e.g. "Podunk". Open it as described in lesson 1 the file "global.cfg" in this directory and change the names below [name] and [friendlyname] also to "Podunk".

Now start the editor, answer the question now with "No" and choose "global.cfg" from the subdirectory "Podunk".

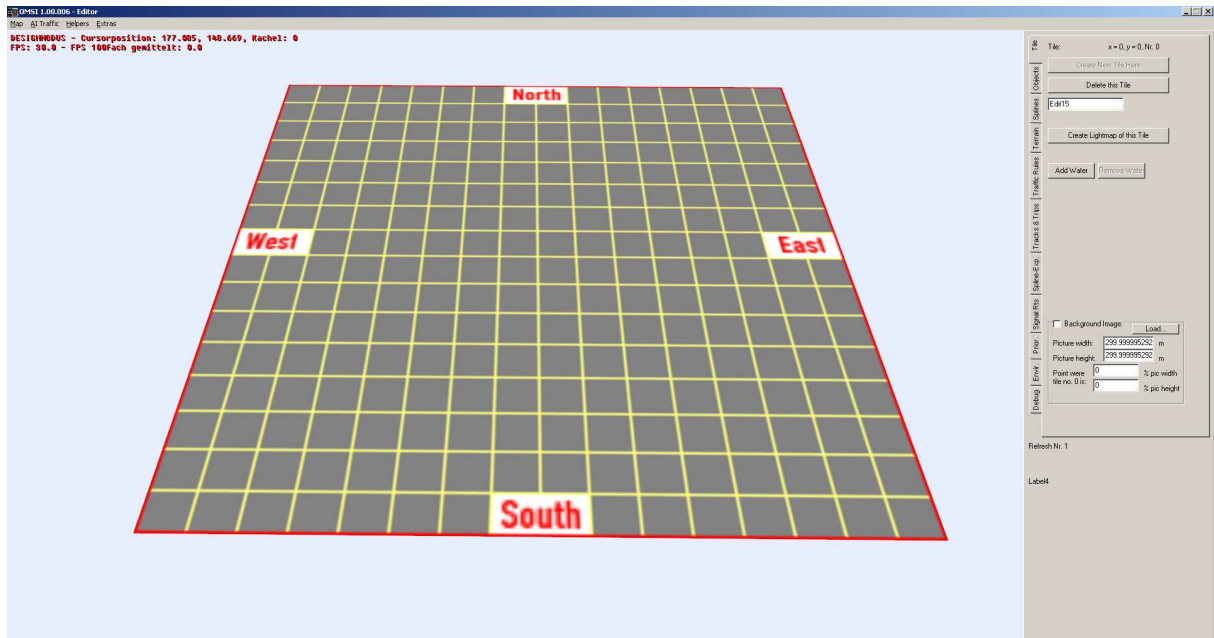
Since this map was copied from an empty template, you can see just one tile without any object:



Now we will try some tile and terrain functions first.

3.2. Tiles

Every OMSI map consists of several tiles of 300 x 300 meters like a puzzle. Each tile is an own file. For a better understanding, click on „Helpers“ => „Show Grid“ in the menu:



As you can see, you have exactly one tile. This tile has a special property: It is the "null tile" in the map's origin, so it has the coordinates "0/0".

3.3. Background Image

At this point, I just want to tell you something about the background image function (without example, it is not too complicated and if you have questions, just ask in the forum!):

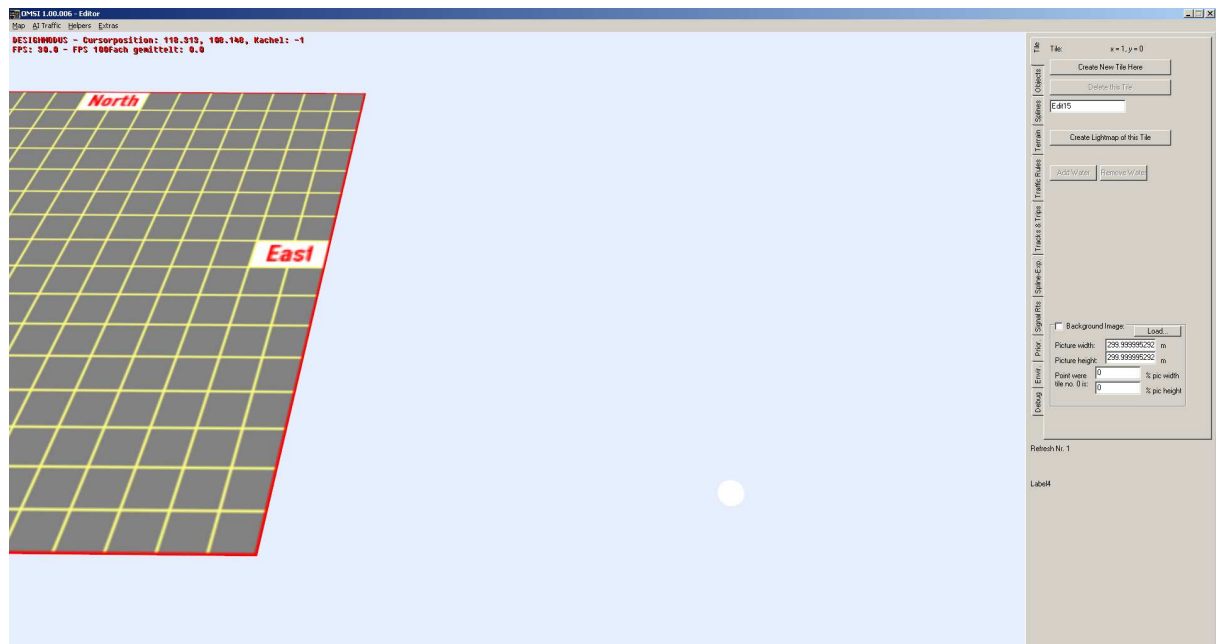
You have to know the following information about your background image: Height and Width in meters (north-south/east-west size) and the point on the image where the null tile has to be positioned, measured in percent of image width/height from the north-west corner.

Then you just need to change to the tab "tile", click in the box "Background Image" on the button "Load..." and choose your image. Then just enter the mentioned numbers. Please mind: The image will only be displayed in the map editor.

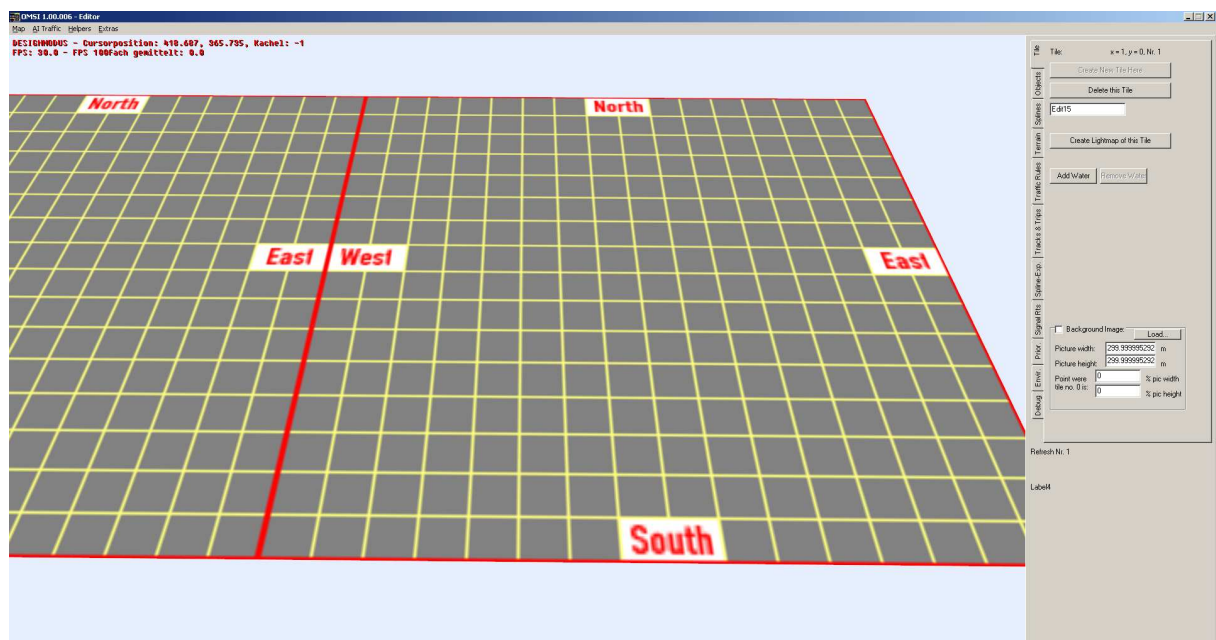
3.4. Add Tile

Now back to lesson: This one tile is too small. We want to add another tile:

Click with the right mousebutton on the "empty area" next to the existing tile, so the map camera will center there (if this does not work, change to the "Objects" tab and back to the "Tile" tab):



On the top of the tab you now can read "Tile x = 1, y = 0", these are the tile coordinates where the map cam is centered currently. Now click on "Create New Tile Here".

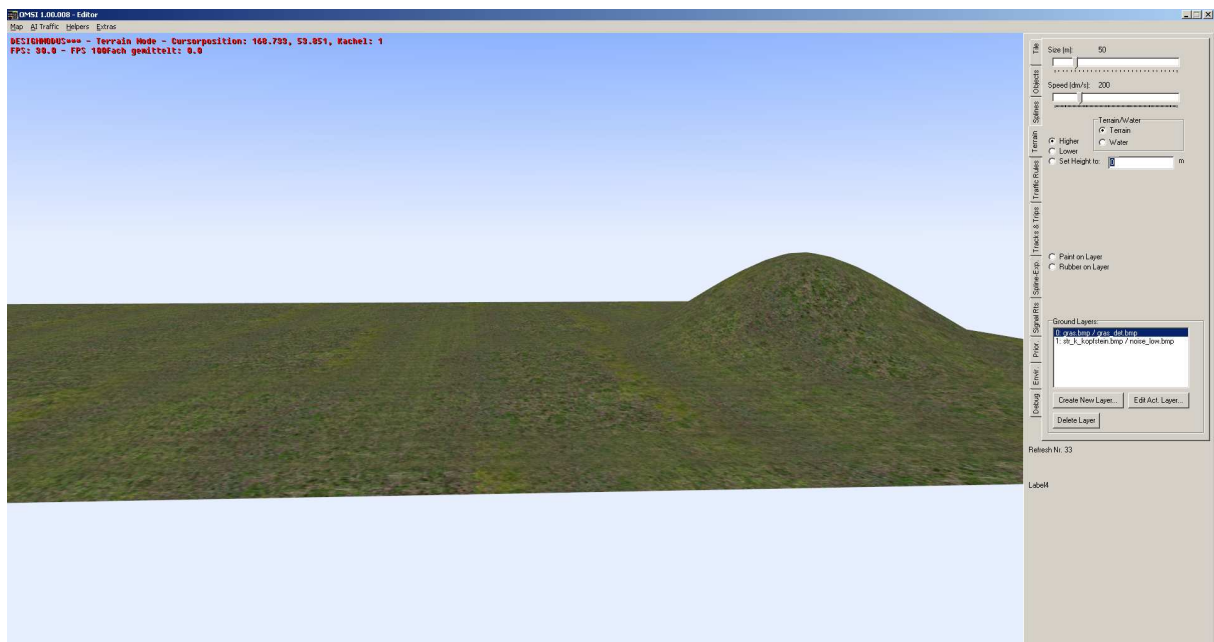


This was it! Now you can read "x = 1, y = 0, Nr. 1", so now there is the tile no. 1.

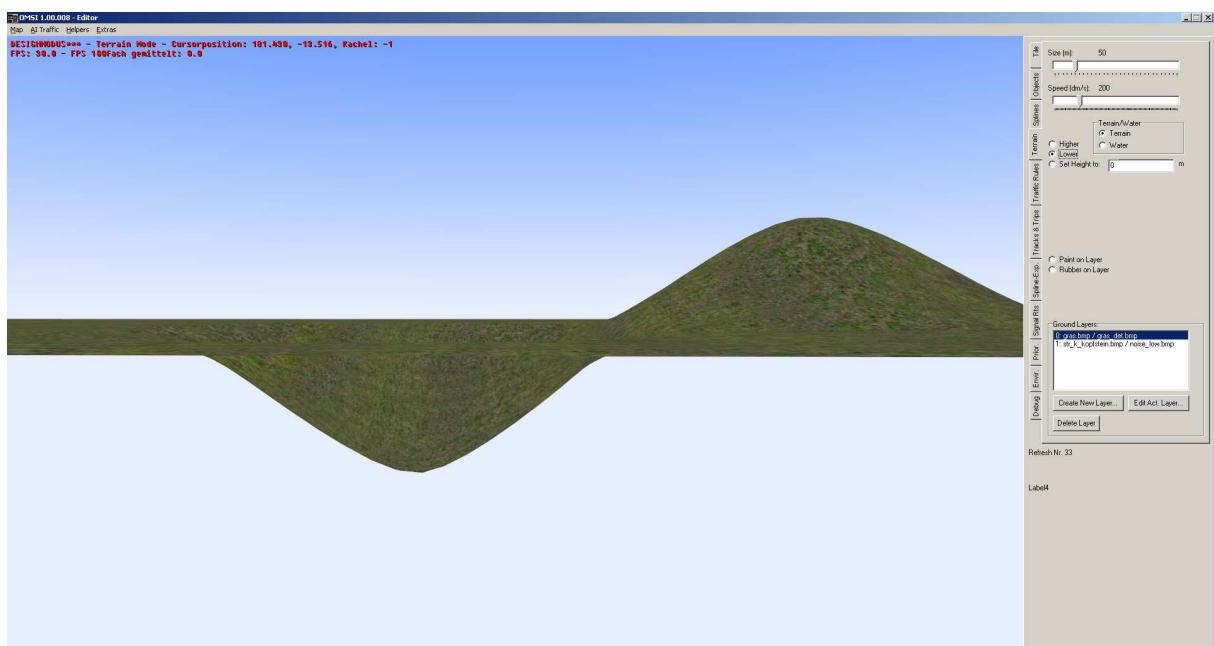
Important: There is no function for tile deleting implemented yet... we have not needed it until now! ☺ The button "Delete this Tile" is not enabled yet. Hopefully we will change this problem with a greater update of the map editor... (it is not the only bug as you know...)

3.5. Editing Terrain

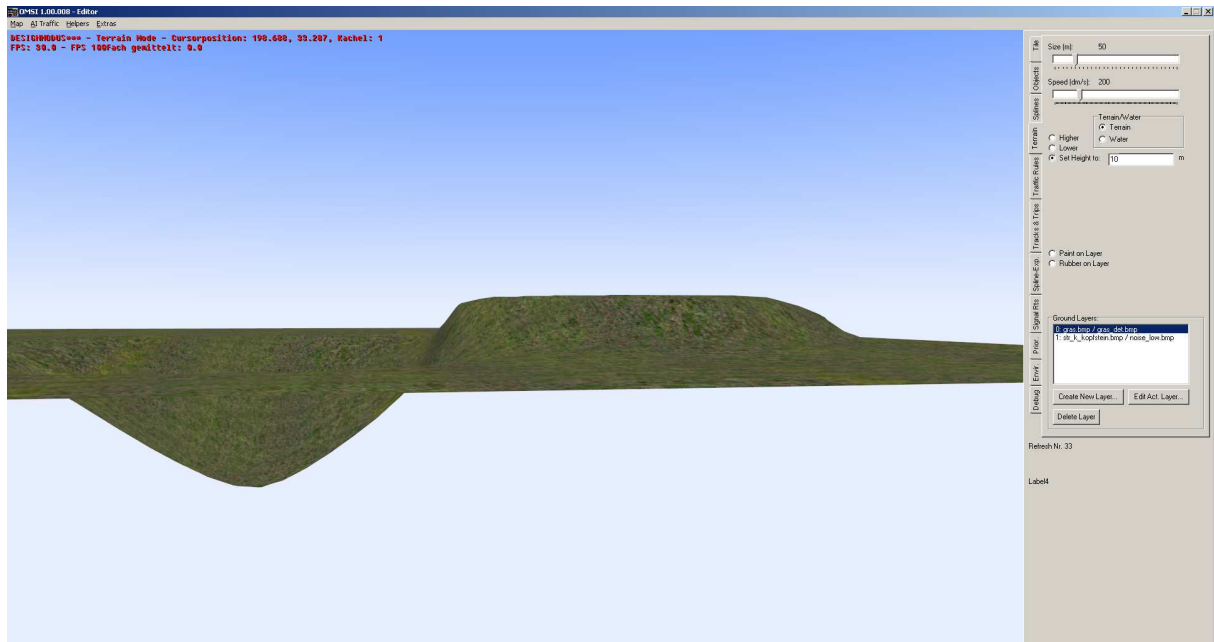
Now change to the tab "Terrain". Leave "Size" on 50, "Speed" on 200, "Terrain/Water" on "Terrain" and "Higher". If you now paint over the terrain, you will raise it there!



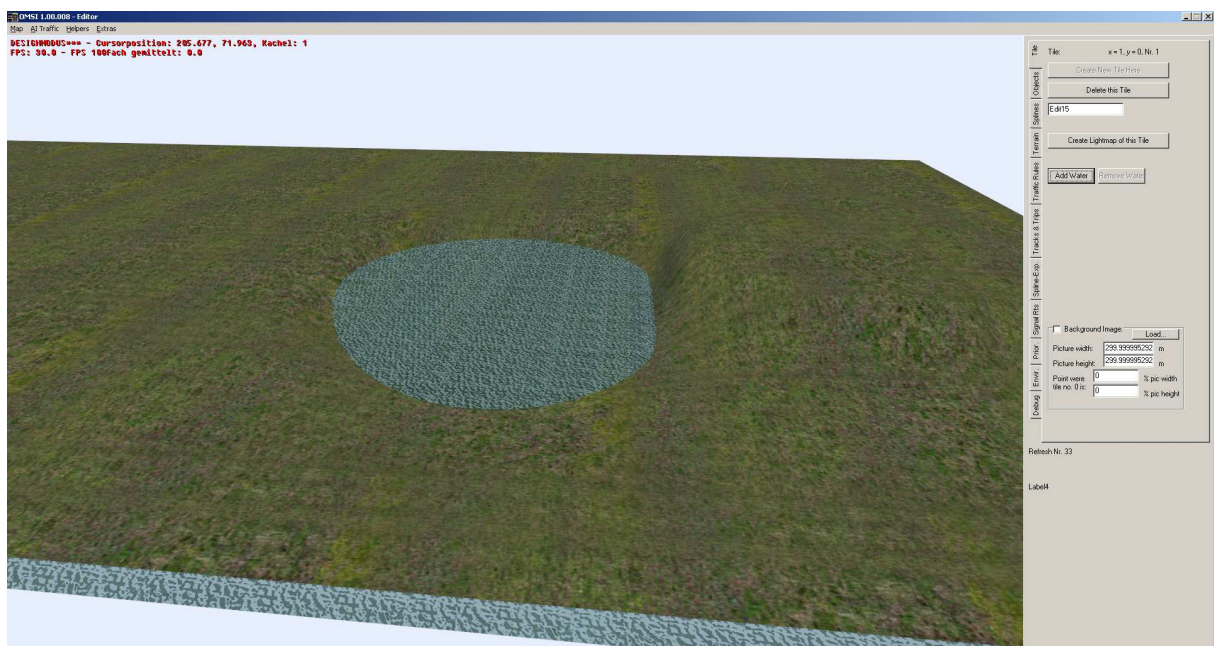
The other way round, you have to use "Lower":



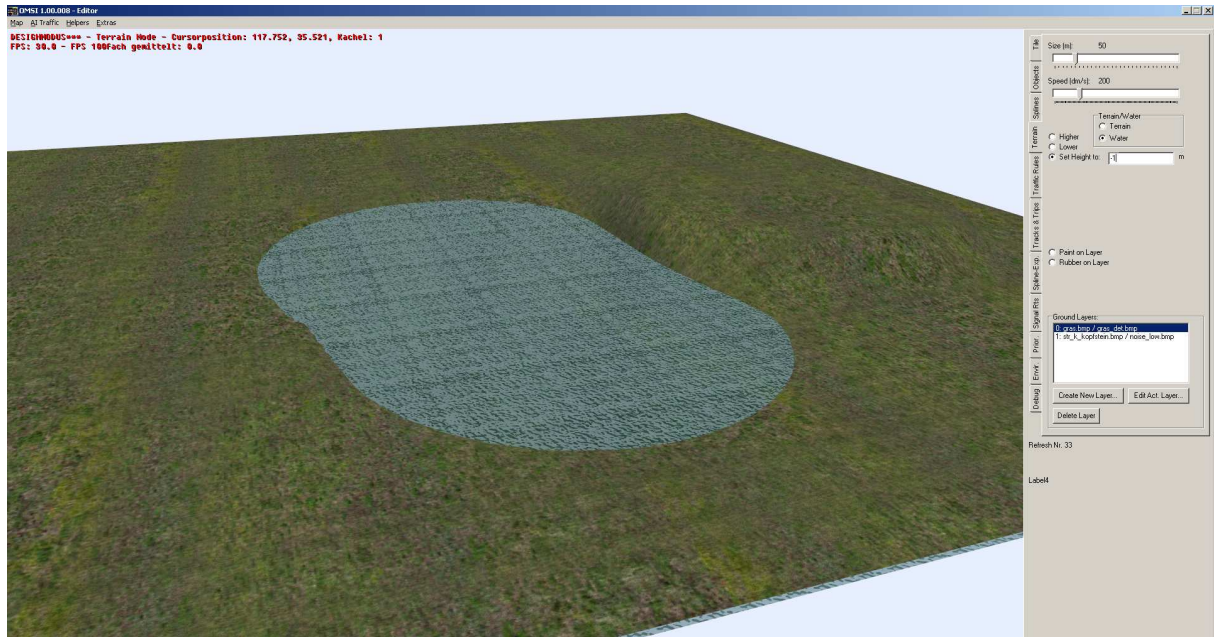
If you choose "Set Height to:", you can enter a special height, e.g. 10 m:



Now we have a nice hole... so we want to fill it with water: Click while centering the map cam on the right tile, on tab "Tile" on the button "Add Water":



Now you can use the same terrain editing functions we have used above, but in mode "Water". But please mind: There are only forming points at each tile corner! In the middle of the tiles, you cannot change the height of the water. You can now set the water level to -1 if you choose "Set Height to:" and enter "-1".



3.6. Terrain Painting

On the tab "Terrain", you will find the possibilities to paint on the terrain, too. For this purpose, we have add a system of layers:

Let's paint some cobblestones:

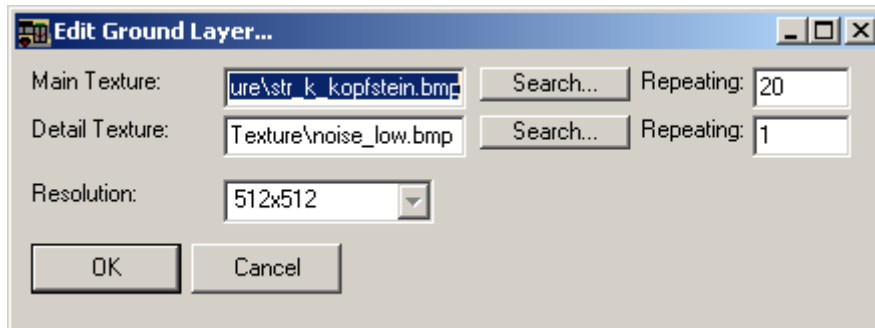
Move the map cam to an unused piece of terrain. Choose "Paint on Layer", "Size" to 2 and choose "Ground Layer" => "1: str_kopfstein.bmp". Now paint on the terrain:



If you choose "Rubber on Layer", you can rubber the painted areas – but in *this* layer only!

Of course, you can add further layers, edit the existing ones or delete them (but of course not the layer 0).

Click on "Edit Act. Layer..." while layer „1: str_kopfstein.bmp“ is selected:



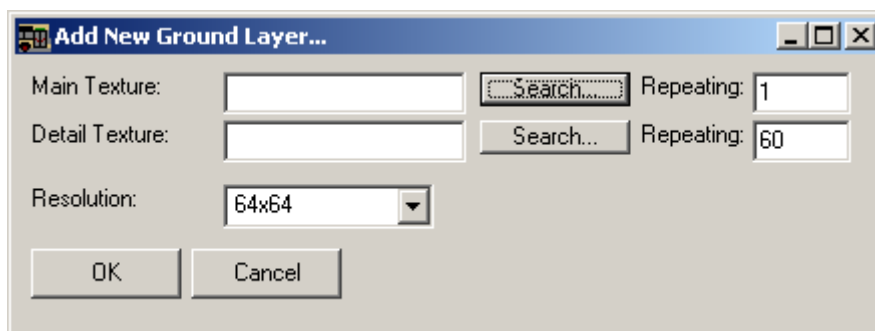
These are the properties of the selected layer. In the first line are the properties of the main texture, filename and the repeating along the 300m tile border.

In the second line the same properties of the detail texture (in this case, it is used as non-repeating, large overlay).

Below that you find the resolution of the mask texture. These textures (cobblestone, asphalt and so on) should have a larger resolution like 512x512, else they could be too unsharp. Other textures like dirt could have lower resolutions.

Important: The resolution cannot be changed after the layer was created! So you have to decide before using it!

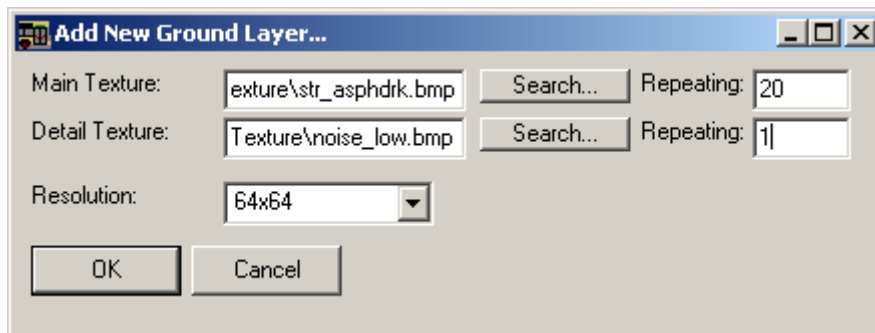
Now click "Cancel" and then "Create New Layer..."!



Click on "Search" next to "Main Texture" and look for the file "str_asphdrk.bmp" in the directory "texture" and enter a repeating of 20.

Then search for "noise_low.bmp" as detail texture with a repeating of 1.

Leave the resolution on "64x64":



Click onto OK – now paint on the map:



You can see now the significant lower resolution and the layer order: layer 2 is painted after layer 1!

Important: The order cannot be changed (yet)! So you should know the order and layers before starting building the map!

Important: Theoretically, you can have a large number of layer. But please mind: It costs performance if you use all together on a special tile! So either you have only few layers or you use just a few layers per one tile and you should not use too high resolutions.

Lesson 4: Creating a new bus route

Now let's get to the point: We will now create a custom bus route.

In the last chapters you have learned the important functions to create the scenery. Now use them to create your first short bus route!

4.1. *Placing streets and intersections*

At first, change to the "Object" tab, click onto "Load all!" and verify the question by clicking "Yes". After a while, the list will contain all objects installed by OMSI (and custom-made objects, too).

The list is a bit larger than before, but you'll have some experience by now! 😊

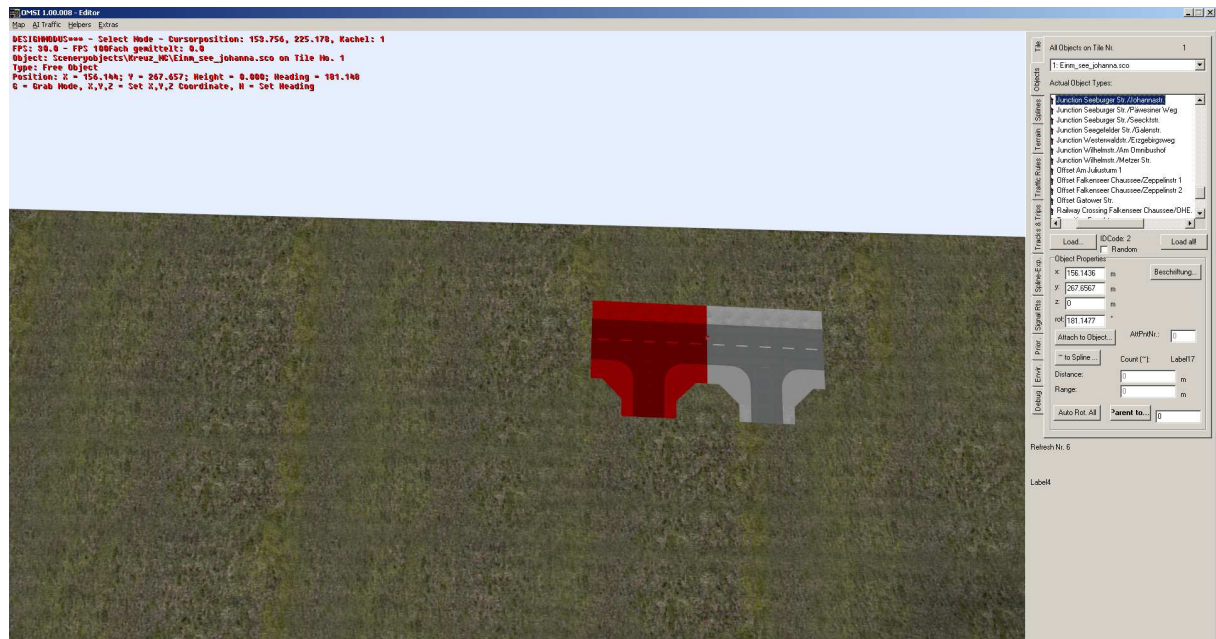
The Spandau map uses a large amount of different intersections. Some intersections and street types can be combined like in a construction kit. Which streets fit which intersections is explained in the appendix.

The main street will be the "Seeburger Str." with a width of 11m. Unfortunately, there is no bus loop fitting that street type. But we can build one from other existing street types and later prohibit unscheduled traffic in the loop.

Search for the intersection „Crossings \ Junction Seeburger Str./Johannastr.“ and place it as seen in the screenshot:



Place another one directly next to it:



After that there will be 200m of street. On the "Splines" tab, click "Load..." and select the file „Marcel\str_2spur_11m_SeeburgerStr1.sli“. Click to open.

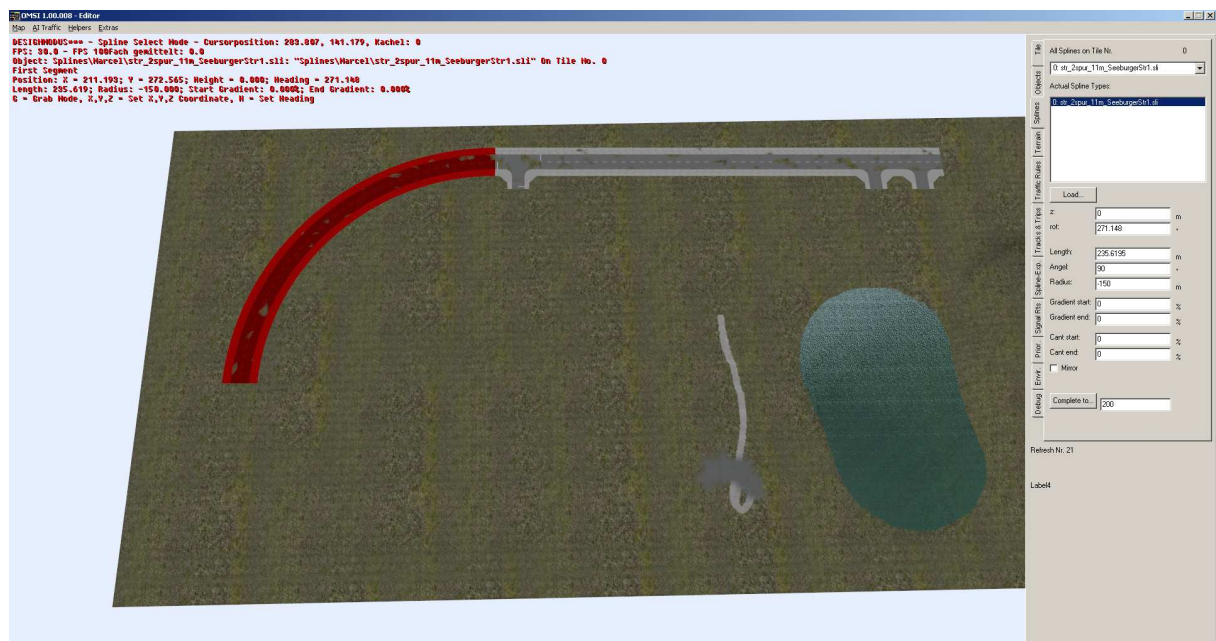
Insert 200m of that spline type on the left:



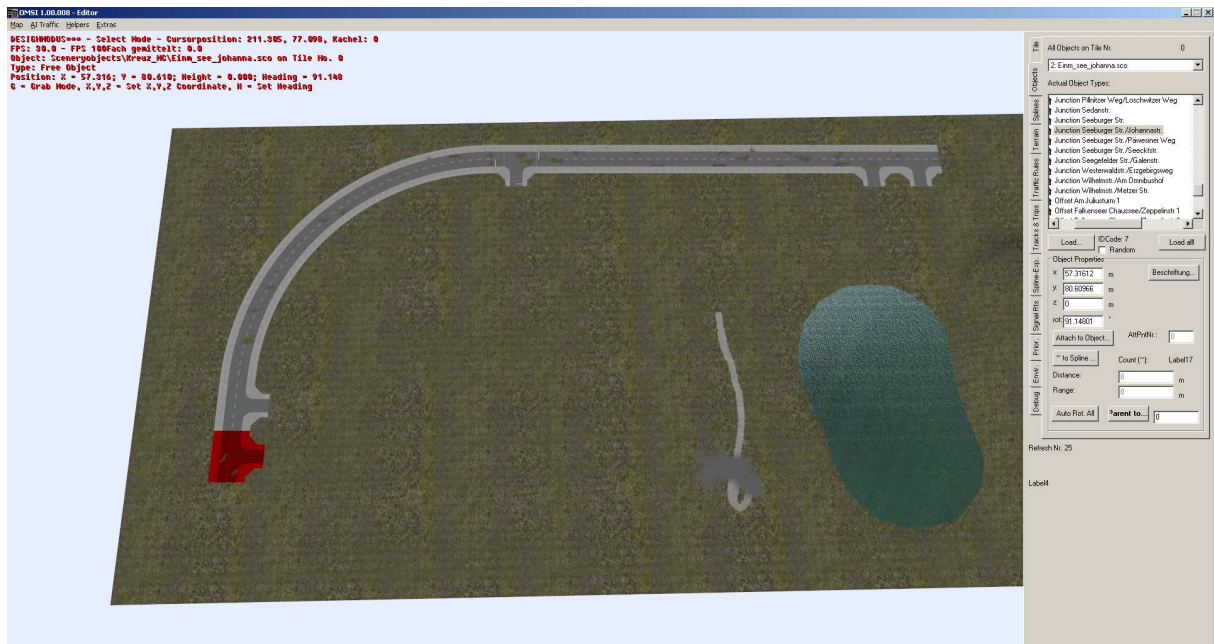
After that there will be another intersection for you'll be able to expand the layout later on! ;-) Use the intersection „Junction Seeburger Str./Päwesiner Weg“ this time for it is suitable for traffic lights:



After that comes another section of street to make the turn. Enter 150m radius at first, if you can manage the turn with that value. Otherwise change the value until it fits or create another tile. You can also enter "90" at "Angle" instead of entering a length. Then you'll have a perfect 90° turn!



At the end, insert another two intersections of the "Johannastr." type:

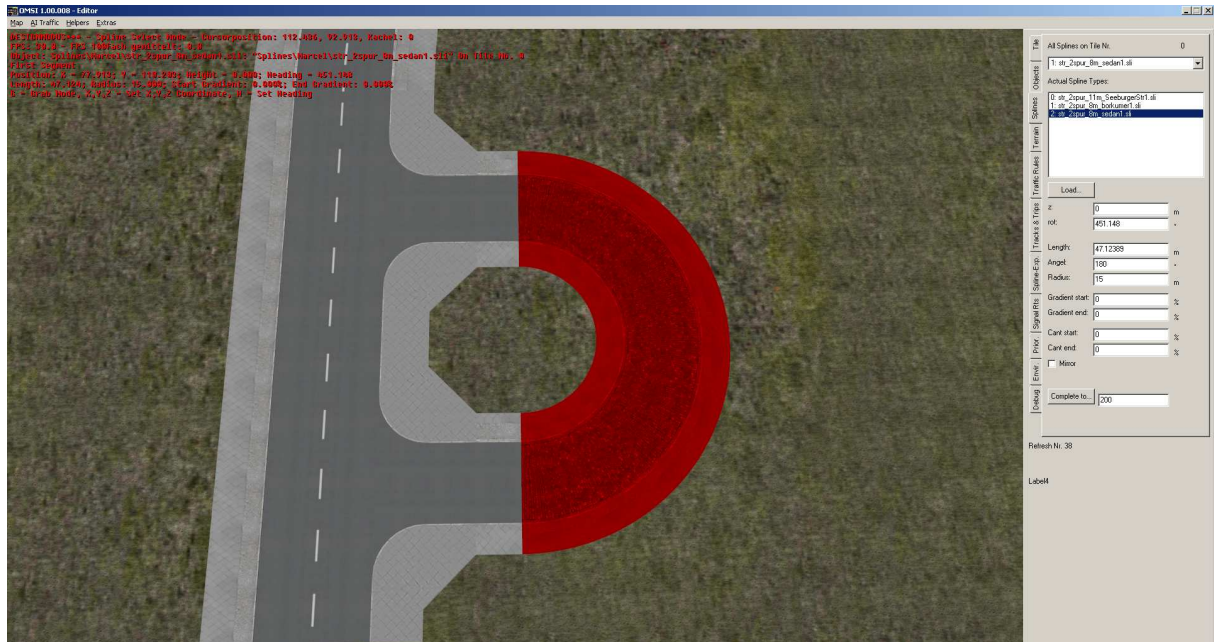


Now let's get to the two loops:

With the automatic completion function mentioned above, you could make OMSI search for the correct solution. But: Angles greater than 179° cannot be found, so you'd have to lay the first piece of road manually anyway.

In this case it is much easier: Our intersections usually have even meter lengths, so you can "guess" the correct radius and then just enter 180° for the angle.

Select the spline „Marcel\str_2spur_8m_borkumer1.sli“ („str_2spur_8m_altonaer1.sli“ (sphalt) or „str_2spur_8m_sedan1.sli“ (cobblestone) are also possible) and insert a new segment:



Now the route is finished for a start.



4.2. Terrain lightmaps & „Envir“ tab

Now let's turn to a short side note on these two matters.

First, please place a row of streetlights along the road. There is a high variety to choose from, e.g. you can use the type „Large Sodium on large whip pole“ verwenden (you'll find it under „German Street Side \ Lights“):



Place some trees, too:



Now change to the "Envir" tab:

Here you find several adjustable parameters to take a look on the map under different conditions.

The slider on the top sets the time, the one below sets the date. Move the lower slider into the very left position:

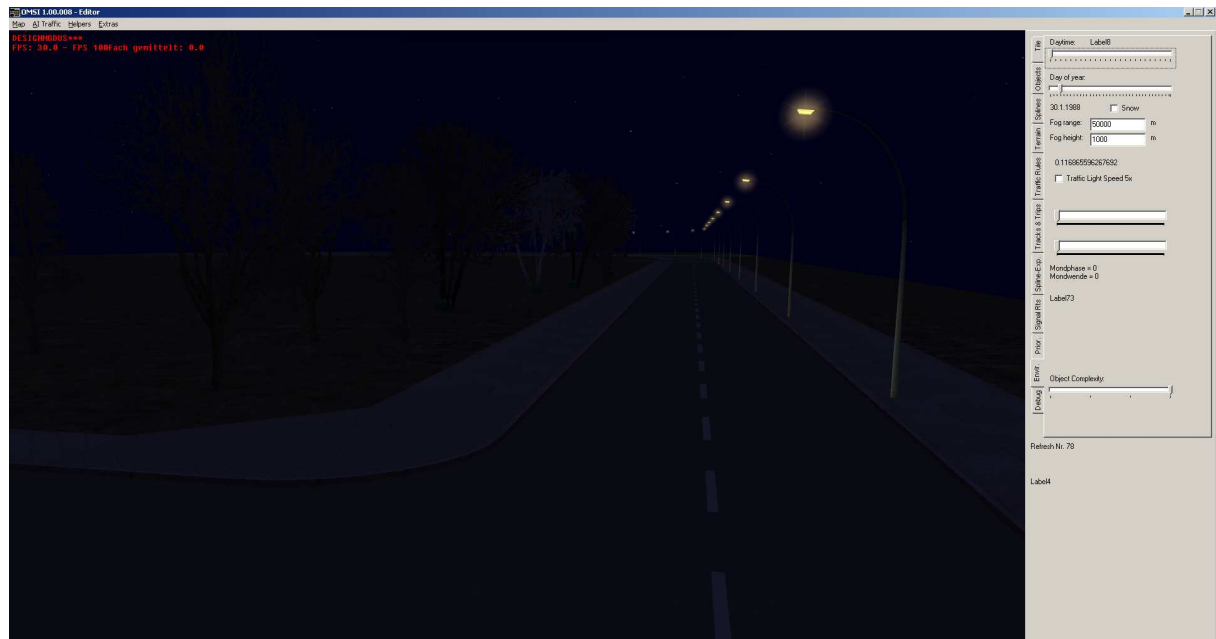


This is what our map looks like during winter! Below there is a checkbox "Snow" you can use to add snow to the scenery:



Important: These parameters will *not* be saved of course, they only serve as a demonstration.

Not turn off the snow again and move the daytime slider to the starting position (night):



As you can see, the streetlights are lit but the street is completely dark!

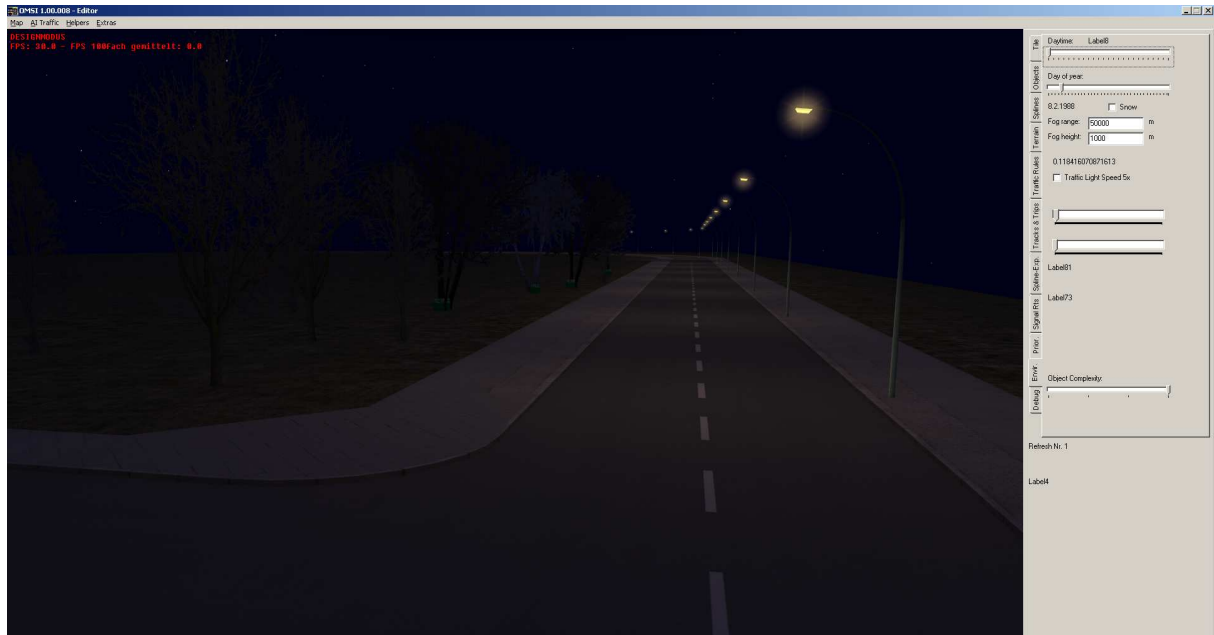
This is where a ner OMSI feature comes along: The so-called Terrain Lightmap! This is a nighttime texture for each tile that can be created automatically by the editor on demand. This texture illuminates streets, objects and the terrain based on the actual light sources.

To achieve a flawless display, Terrain Lightmaps must be created after adding, moving or deleting street lights. On larger maps, this may take some time. Thus, you can create Lightmaps for single tiles, too (there is a button on the "Tile" tab).

In this case, we want to fit all tiles with new Lightmaps. In the "Map" menu, click on "Create lightmap". A dialogue field will appear informing you that the map will automatically saved prior to creating the lightmaps. Click "Yes" to continue.

Before the lightmaps become active, the map has to be reloaded. You can either close and re-open the editor or click "Change" in the "Map" menu to reload the same map.

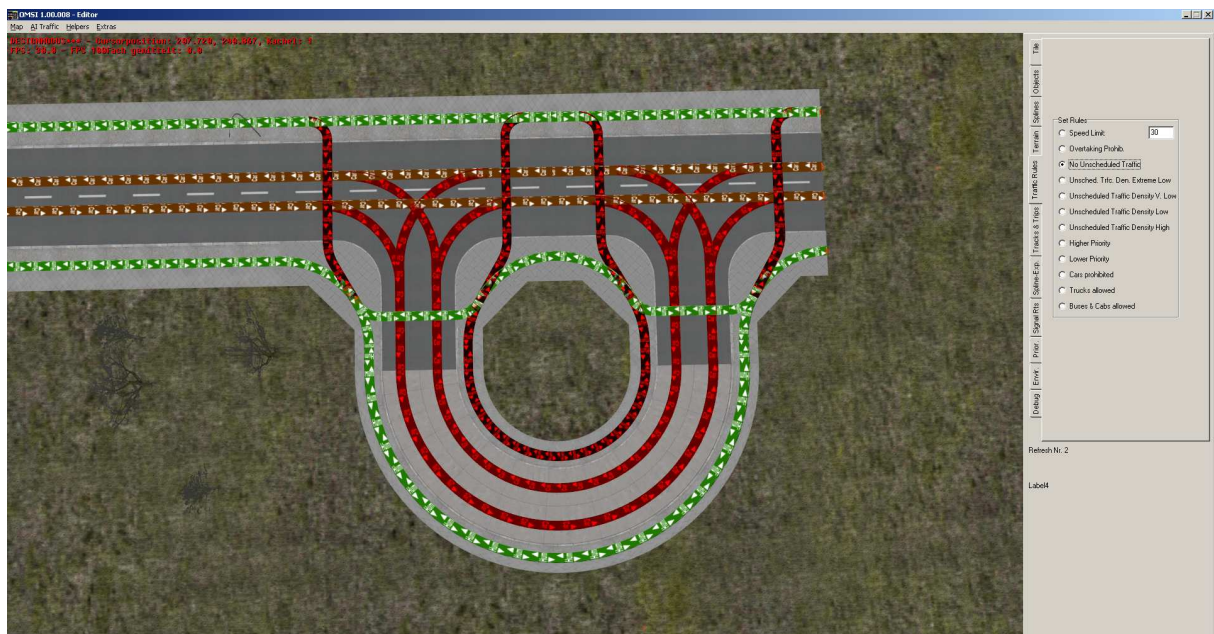
Afterwards, you can set the daytime slider on the "Envir" tab to night again:



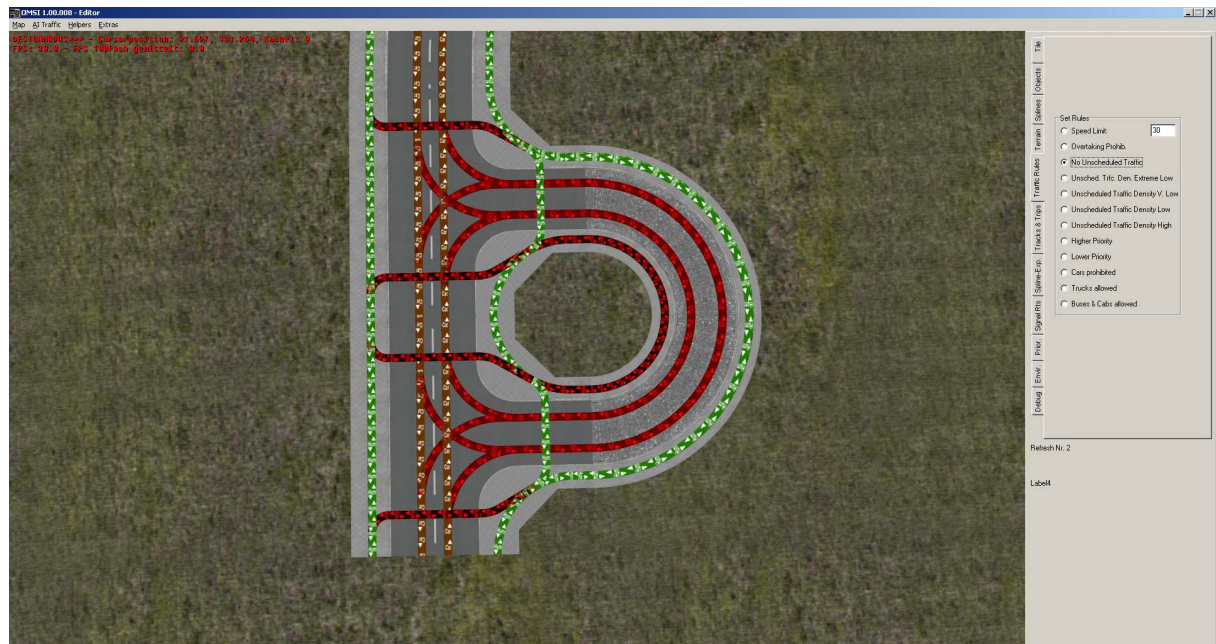
That looks much better, doesn't it? 😊

4.3. *Adjusting traffic rules*

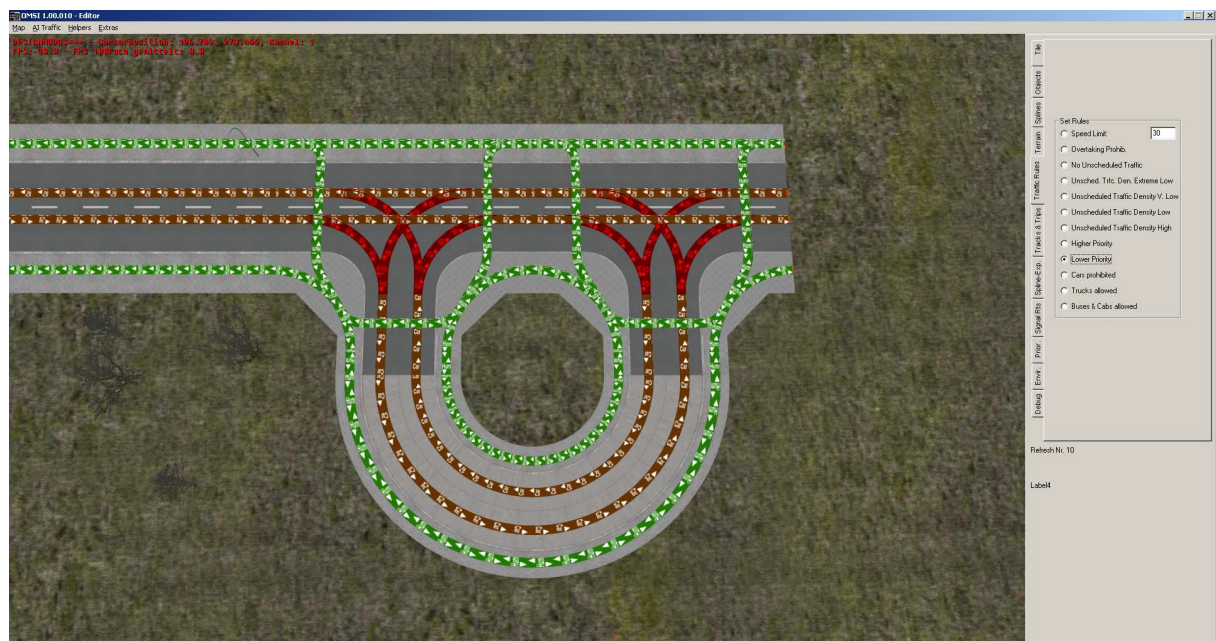
Now the loops need to be closed for normal traffic. Marking them with "No unscheduled traffic" is the best way to do this if no other un'scheduled traffic is supposed to go through the loop. Furthermore, some of the pedestrian paths will be deactivated:



The same applies for the other end:



Furthermore we want to prevent the bus from reserving the exit of the loop while parking there. So we lower the priority of the loop junction. Choose "Lower priority" and mark the following paths in both loops:

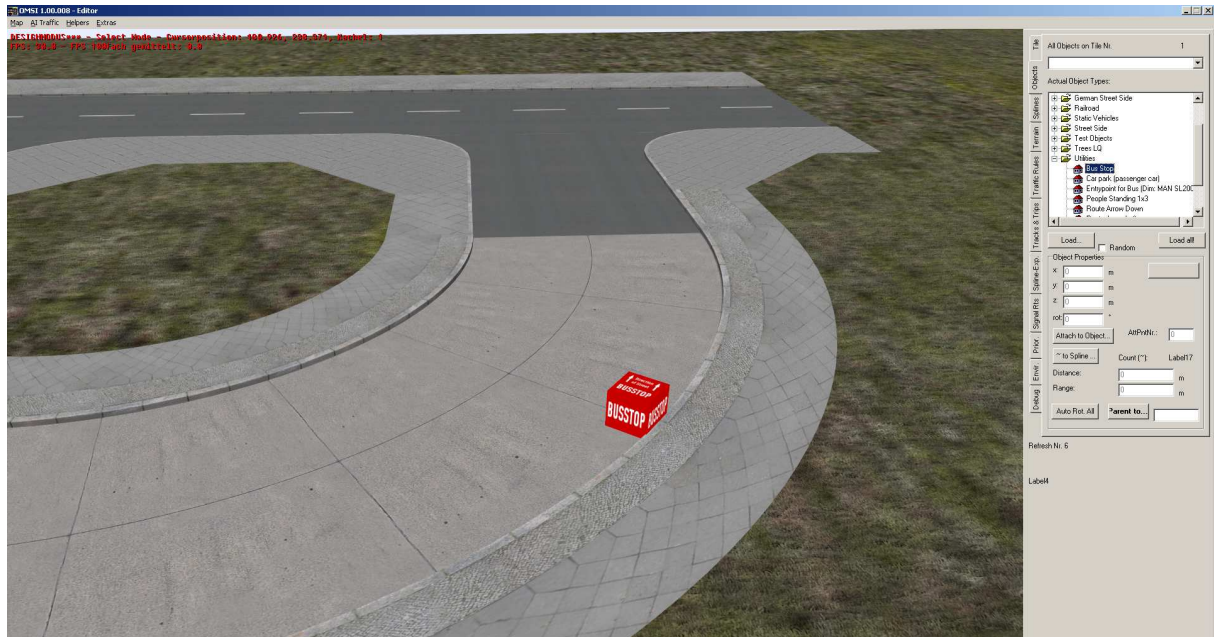


4.4. *Placing bus stops*

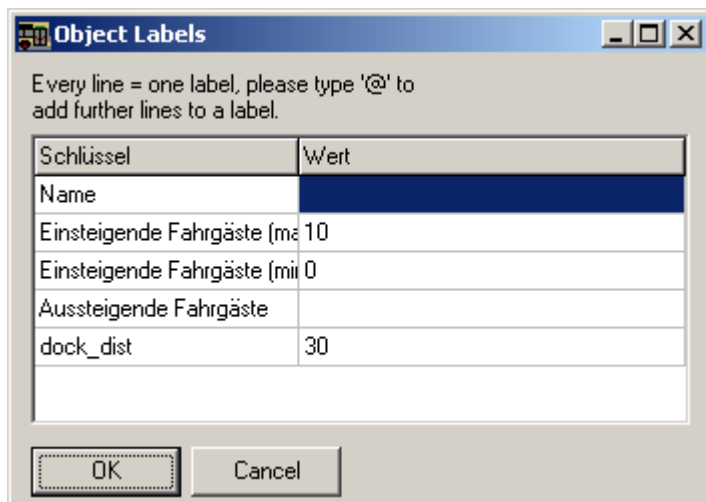
It has been already explained how to place bus stop poles with attached line cards. Now it is time to set them up for real use!

Select the object type „Utilities \ Bus Stop“. This is a red cube labeled "Bus Stop". This is a helper object which is only visible in the editor, marking

the location of the internal bus stop. Place the cube as shown in the northern loop. Please pay attention to the correct direction of the arrows!



Next, we want to edit the bus stop properties. Click onto "Options...":



At first, you can enter a Name. As, by default, the Spandau depot file is assigned to all AI traffic based on the "NewMap" template, we should use the name of a terminus included in the Spandau depot file, so the buses can show this destination later on. We will later show how you can create your own depot file.

Enter the name "Harzer Str" and make sure it is spelled exactly like that (rollsign No. 13, IBIS-code 165).

With the following two values you can set the minimum and maximum amount of passengers waiting at the stop. With this one being a terminus, enter "0" twice.

You can leave the field "Aussteigende Fahrgäste" (number of passengers getting off) empty (then it will be the average of waiting passengers) or, in this case, you can enter a value (which also depends on the number of passengers in your bus). Enter "10" here.

The value "dock_dist" sets the docking distance before the bus stop in meters. Normally you can keep this value. At some stations like Reimerweg, you can use the value to optimize the way Ai buses dock and cast off.

Schlüssel	Wert
Name	Harzer Str
Einsteigende Fahrgäste (min)	0
Einsteigende Fahrgäste (max)	0
Aussteigende Fahrgäste	10
dock_dist	30

Place a second cube where the bus is supposed to take up new passengers:

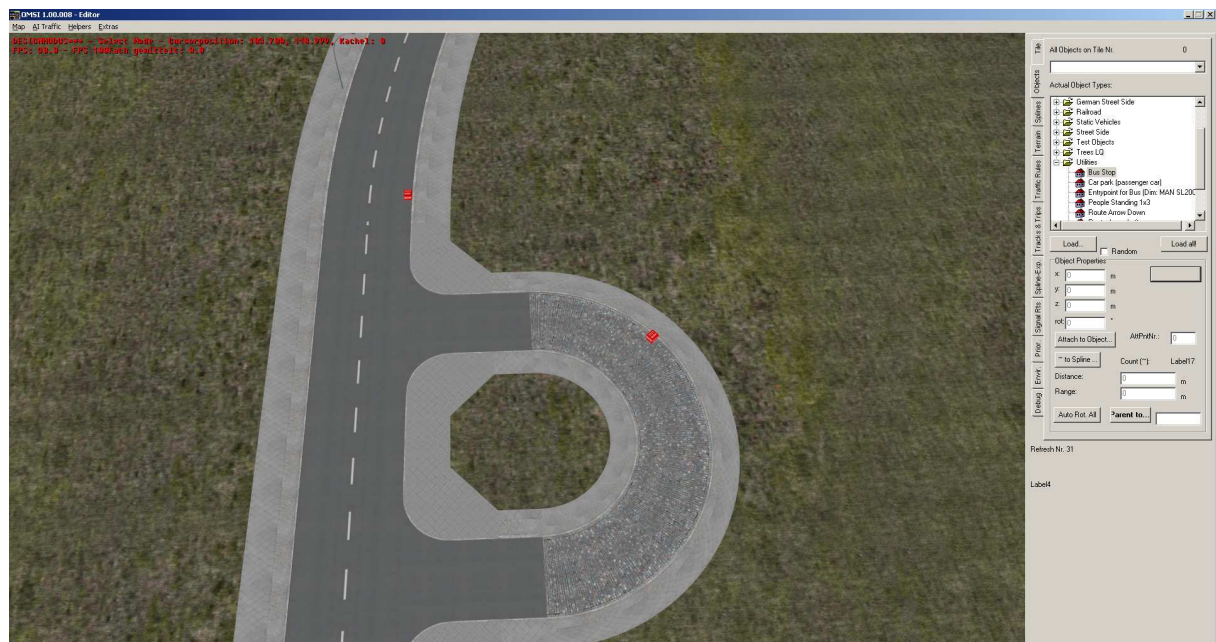


The name of this stop should be the same. However, concerning the waiting passengers, enter a minimum of 2 and a maximum of 10 passengers. The number of passengers getting off should be set to 0.

At the intersection in the middle, two more stops will be places. Let's call them "Goethestr" and assign minimum 0 and maximum 10 waiing passengers and 10 passengers getting off.



At last, another two cubes will be placed at the southern loop. Due to the different shape of the loop, the location of the starting point must be changed:



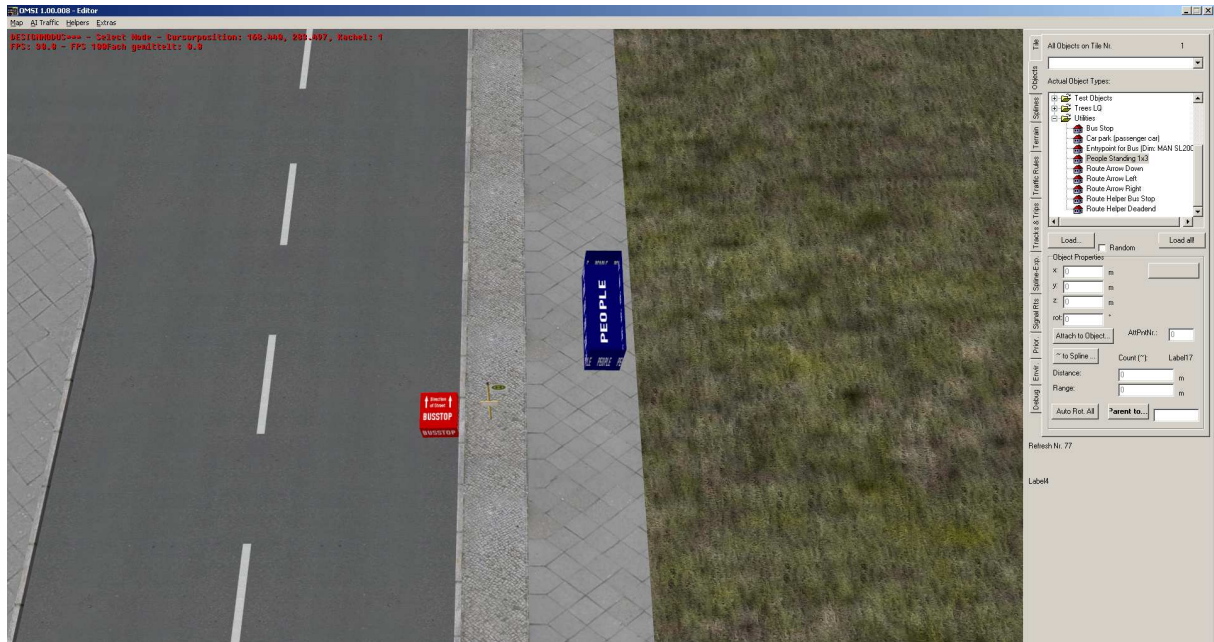
The waiting stop in the loop has 0 waiting passengers (min and max) and 10 passengers getting off. The other cube further north has min. 2 and max. 10 waiting passengers and no passengers getting off. This stop will be names "Brixpl", rollsign No. 16, IBIS-code 168.

Next, we will place the bus stop signs. Use the „Bus Stop with name plate and 1 timetable“ for the regular stops and „Bus Stop Terminus“ for the stops inside the two loops. You already know how to label them. If you want, you can add some line cards for the new line "104". With this number having three digits, you have to use the line card „Bus Stop Linecard yellow 3-digit“.



Attention: The labels entered for the red cubes are the "official" names of the bus stop! They are used in the timetables, in the depot files and to recognise the terminus. On the other hand, it doesn't matter what you write onto the visible bus stop signs. Of course it should match so the human driver knows, too.

The waiting positions for the passengers are still missung. Place some of the objects called „Utilities \ People Standing 1x3“ next to all stops where people can get on the bus. Make sure (label!) the box is oriented correctly so that the waiting passengers will look in the direction of the arriving bus.



Furthermore, you can place a waiting hut which even has seats! („German Street Side \ BVG \ Bus Stop Hut“) These can be labeled, too, and don't forget to set the correct height (0.25m).



Now place waiting positions on all necessary stops.

Attention: As long as there is no timetable in the map (as it is right now), there will always be passengers waiting at the stop. Once you have defined a timetable, passengers will only wait there if there is a scheduled departure within the next minutes.

Before you start your first test run, you should place an entry point in the northern loop and label it "Harzer Str.".



Now start OMSI and test your first bus route! You will notice that the passengers will already react to the destination shown on your bus (reminder: Brisplatz = rollsign No. 16, IBIS-code 169, Harzer Str. = rollsign No. 13, IBIS-code 165).



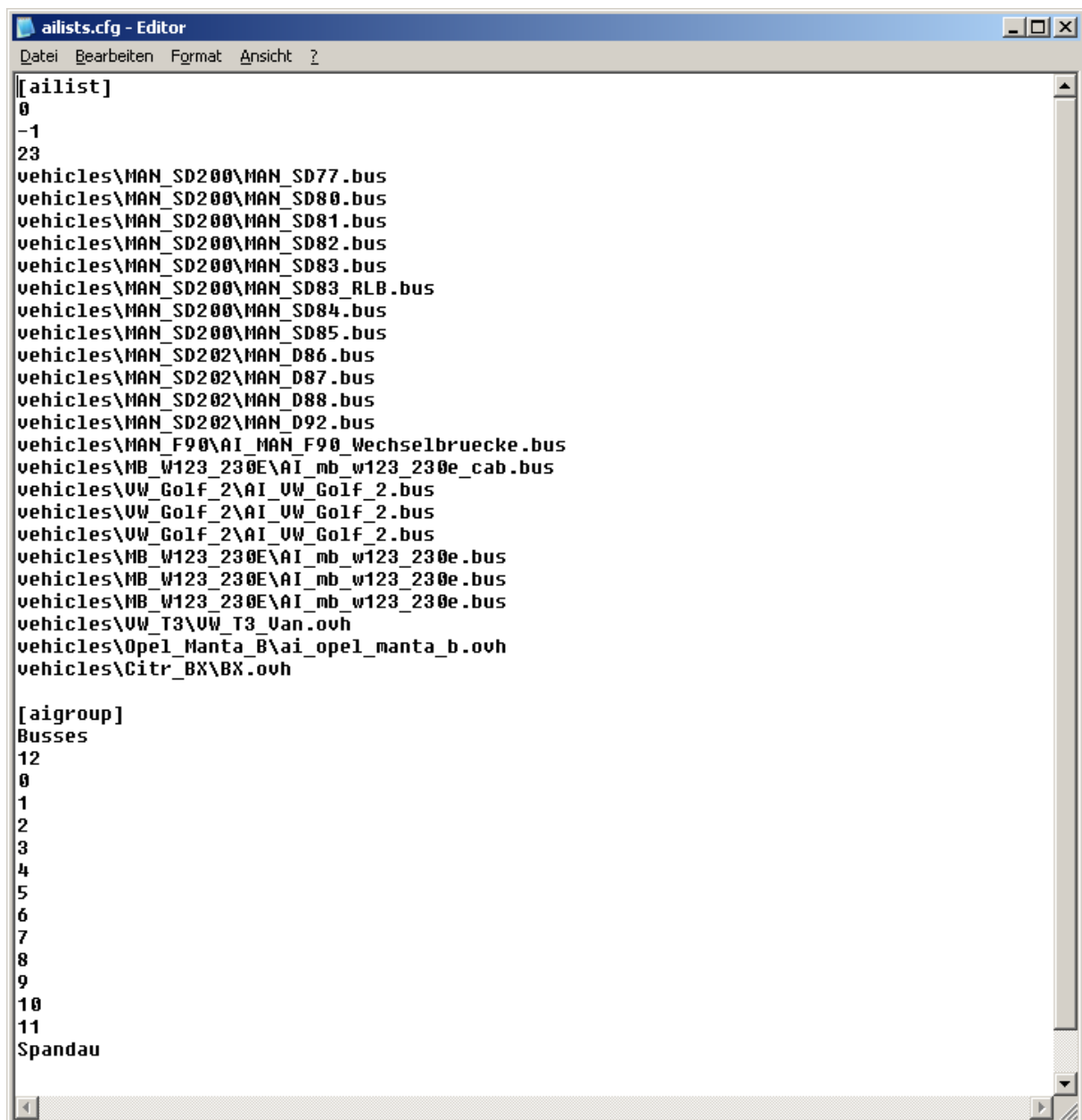
Lesson 5: Scheduled AI traffic

In the last map editor lesson we will create a timetable for our created bus route so that scheduled AI traffic will be possible, too.

But first we will take a closer look at the file "ailists.cfg" which assigns the different vehicle types.

5.1. Introduction to configuration files

Open the file "ailists.cfg" from the "maps\Podunk" directory with Wondows Notepad. You have learnt how to do that in the beginning, creating the map "myMap".



```
[ailist]
0
-1
23
vehicles\MAN_SD200\MAN_SD77.bus
vehicles\MAN_SD200\MAN_SD80.bus
vehicles\MAN_SD200\MAN_SD81.bus
vehicles\MAN_SD200\MAN_SD82.bus
vehicles\MAN_SD200\MAN_SD83.bus
vehicles\MAN_SD200\MAN_SD83_RLB.bus
vehicles\MAN_SD200\MAN_SD84.bus
vehicles\MAN_SD200\MAN_SD85.bus
vehicles\MAN_SD202\MAN_D86.bus
vehicles\MAN_SD202\MAN_D87.bus
vehicles\MAN_SD202\MAN_D88.bus
vehicles\MAN_SD202\MAN_D92.bus
vehicles\MAN_F90\AI_MAN_F90_Wechselbruecke.bus
vehicles\MB_W123_230E\AI_mb_w123_230e_cab.bus
vehicles\VW_Golf_2\AI_VW_Golf_2.bus
vehicles\VW_Golf_2\AI_VW_Golf_2.bus
vehicles\VW_Golf_2\AI_VW_Golf_2.bus
vehicles\MB_W123_230E\AI_mb_w123_230e.bus
vehicles\MB_W123_230E\AI_mb_w123_230e.bus
vehicles\MB_W123_230E\AI_mb_w123_230e.bus
vehicles\VW_T3\VW_T3_Uan.ovh
vehicles\Opel_Manta_B\ai_opel_manta_b.ovh
vehicles\Citr_BX\BX.ovh

[aigroup]
Busses
12
0
1
2
3
4
5
6
7
8
9
10
11
Spandau
```

Configuration files are all those files describing maps, buses, wheather scenarios and so on. They usually have the extension "*.cfg", but also "*.ovh" or "*.bus" for vehicle files, "*.sco" for scenery objects or "*.sli" for spline types. But all of them are configuration files.

Some basics first, which are valid for all OMSI configuration files:

There are so-called "keywords" which are essential! These keywords vary between the different configuration files. So the "ailists.cfg" contains other keyword than the previously opened „global.cfg“.

Only if OMSI finds such a keyword, the following lines will be imported (depending on the structure of the expected set of data). Having read all lines of data belonging to a certain keyword, OMSI starts to look for new keywords again.

As long as OMSI is searching for the next keyword, all other things will be ignored and be treated as comments.

Important: Keywords must be spelled correctly!

- Case sensitivity must be obeyed
- No space characters are allowed, neither before, nor after, nor between keywords
- In the line of the keyword, *there must not be written anything else!* Not even after the keyword.

Let's now have a look at the open "ailists.cfg". Each map can have such a file. In this file, the vehicles participating in the AI traffic are defined. Furthermore, the functions "Random bus" and "Show only numbers fitting to the map's depot" are based on a list contained in this file

At first you will find the entry "[ailist]". This is a keyword! After that there are the following parameters:

0 = vehicle class. This value is not used, it should always be zero

-1 = vehicle group that shall be used for unscheduled traffic. With this number being "-1", all vehicles that don't belong to a vehicle group, will be used for unscheduled traffic.

23 = number of the following vehicle entries.

Now you will see a list of vehicles.

The next keyword is "[aigroup]" defining a group of AI vehicles:

„Busses“ = Name of the group

12 = number of vehicle types in the group

0...11 = index numbers from the [ailist], i.e.. 0 = MAN_SD77.bus, 1 = MAN_SD80.bus etc.

„Spandau“ = Depot file that will be used for these buses.

All other vehicles like the Golf, the Mercedes or the van have not be mentioned in any group. Thus, they automatically belong to group "-1" so they will be used as unscheduled traffic.

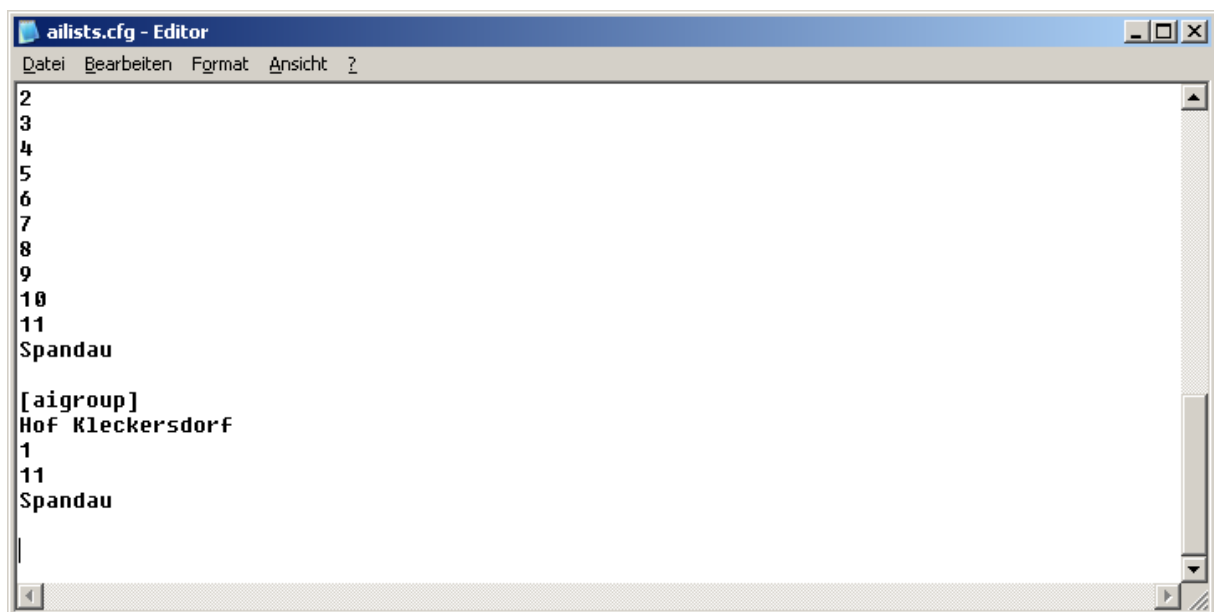
5.2. Adding a new AI group

Let's create a new AI group that only contains buses of D92 stock!

Add the following entry at the end of the file, after "Spandau" and an additional line break:

```
[aigroup]
Hof Kleckersdorf
1
11
Spandau
```

So that's a group named „Hof Kleckersdorf“ (Podunk Depot) with only one type of vehicle: Index 11 = MAN_D92.bus:



5.3. Creating the first scheduled AI trip

First of all: The map editor has shown room for improvement on different occasions, creating timetables will add another weak point.

Important: Timetables should only be created once the streets are finally laid out! If the street layout has to be changed after creating timetables, all timetable data need to be deleted and regenerated. The OMSI editor may crash while trying to edit timetables that use streets which don't exist anymore!

So at first you should build the streets and the crossings. Before you continue creating a large timetable, the layout should be tested with a simple experimental timetable that can be deleted without a guilty conscience afterwards.

And the second drawback:

Important: Deleting lines, tracks and trips (what they are exactly will be explained in a moment) can't yet be deleted in the editor! You will find the files in the "TTData" directory inside the current map directory. They can be manually deleted. But be careful - or, when in doubt, just delete all files in that particular directory. Otherwise there might still be some lines trying to access already deleted tracks or trips which will lead to an editor crash.

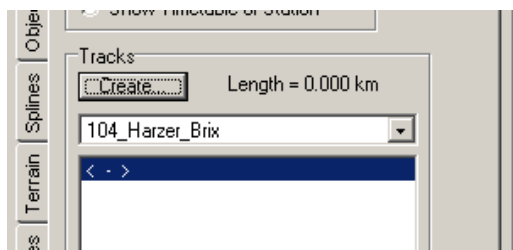
Now change to the "Tracks & Trips" tab which is used for the whole aspect of timetable-related design. In this tab, the paths are visible again.

At first, we'll have to create a "track". That is the section from one terminus to the other. At terminuses, these tracks must overlap each other with at least one path segment, so the bus will be able to change from one track to the other.

Now let's create a new track: At the top, in the "Tracks" section, click onto "Create..." and enter the name "104_Harzer_Brix":



Now make sure that the "Add Trackentry" mode is still active. First, click onto the first line "< - >":



Then click onto the first segment:



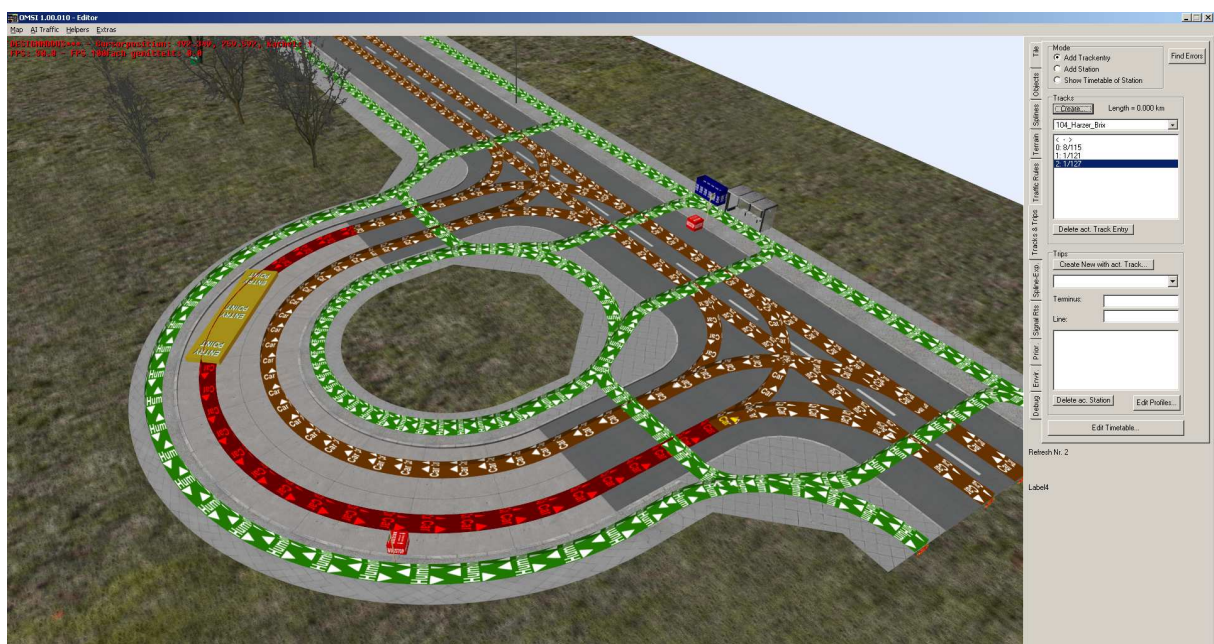
As you can see, this segment is now painted yellow. Now continue with the next segment:



The previous segment is now red, the current one (marked in the list) is yellow.

Important: The order in which you click onto the segments does matter of course!

Now click through all segments up to the terminus. Make sure to click the segments in the correct order. You can remove the current (yellow) segment from the list by clicking „Delete act. Track Entry“ beneath the list.





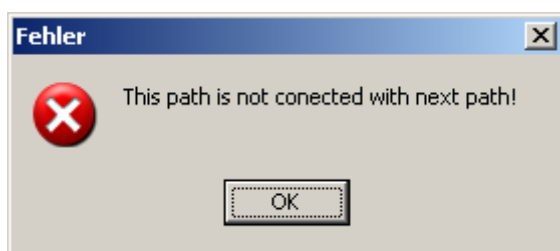
And so on... up to the last segment at Brixplatz terminus:



You can check if you have done everything right! Click onto "Find errors". Ideally, the following report will appear:

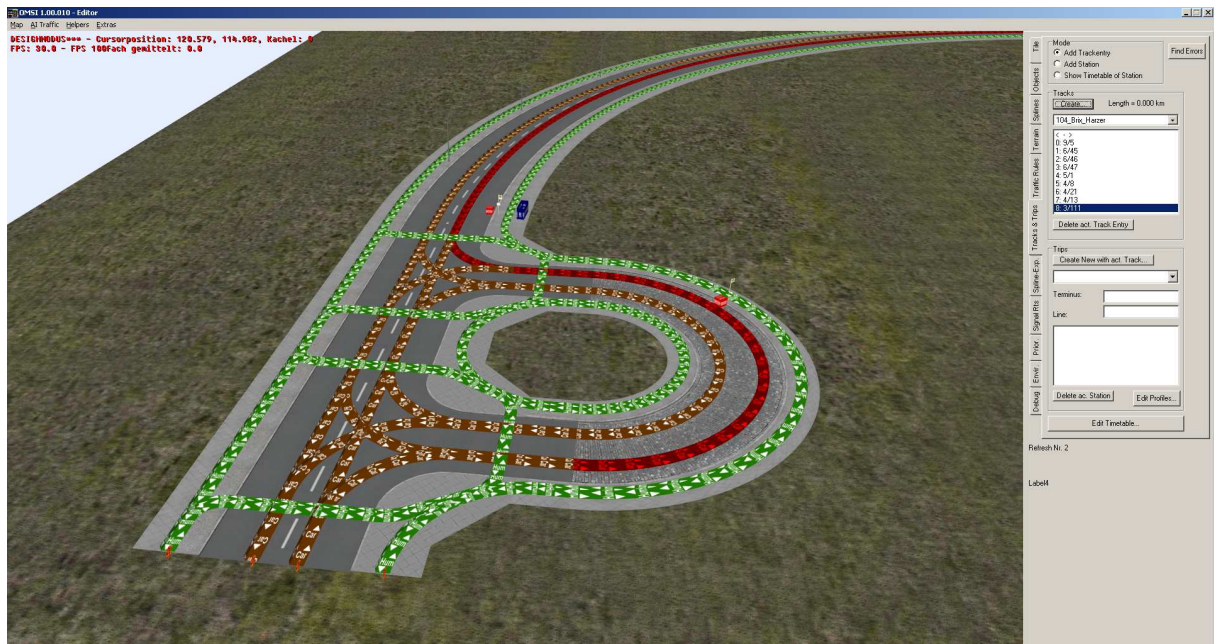


If not (e.g. if you have forgotten a segment inbetween), this box will show up:



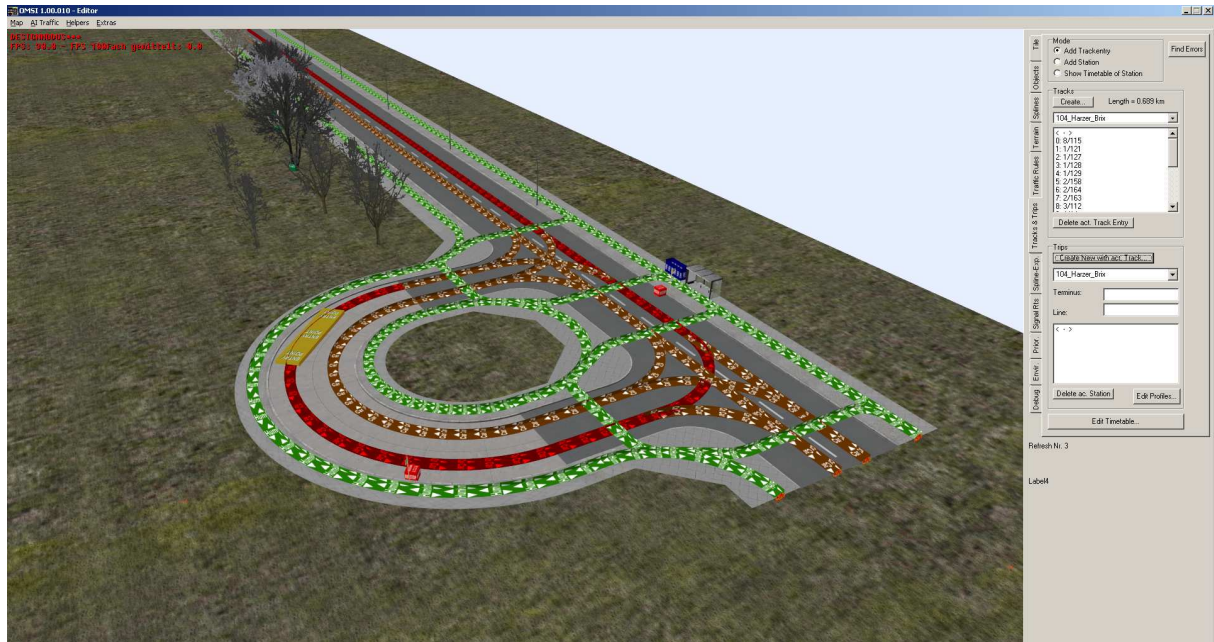
In this case, the segment before the missing segment will be automatically selected, so you'll just need to click onto the missing path segment (if there was only one missing segment). If the order of segments is wrong or similar problems occur, you might need to remove segments from the list.

Create the opposite track „104_Brix_Harzer“ in the same manner:



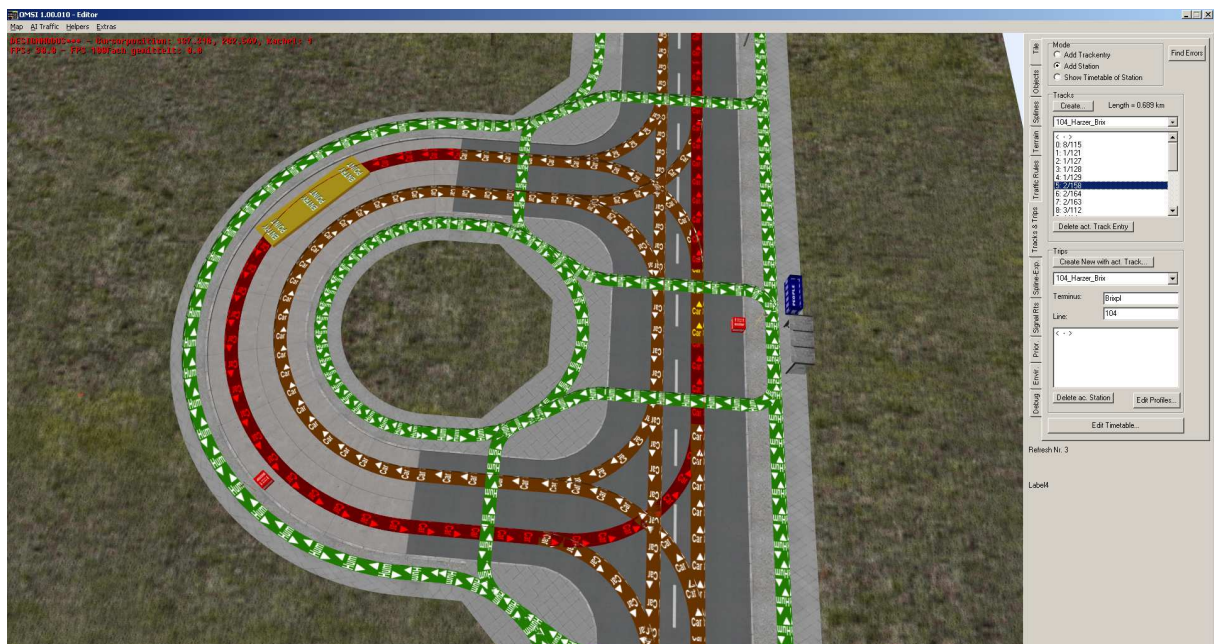
Next, we'll have to create so-called trips. A trip always consists of a track, an additional list of stops, a destination display, a line and one or more profiles. A profile is a list of passing times for each bus stop on the trip. E.g. there can be a profile for rush-hour traffic and another night-time profile (when the bus can advance faster).

But first, let's get back to the two required trips: Select the track „104_Harzer_Brix. Then click onto „Create New with act. Track...” in the "Trips" section below and enter „104_Harzer_Brix" as the name of this trip.

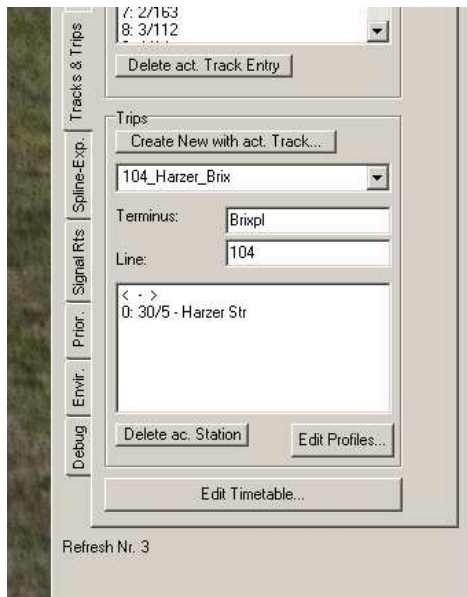


Enter "Brixpl" (spelling!) for terminus and "104" for the line. Now we will add bus stops:

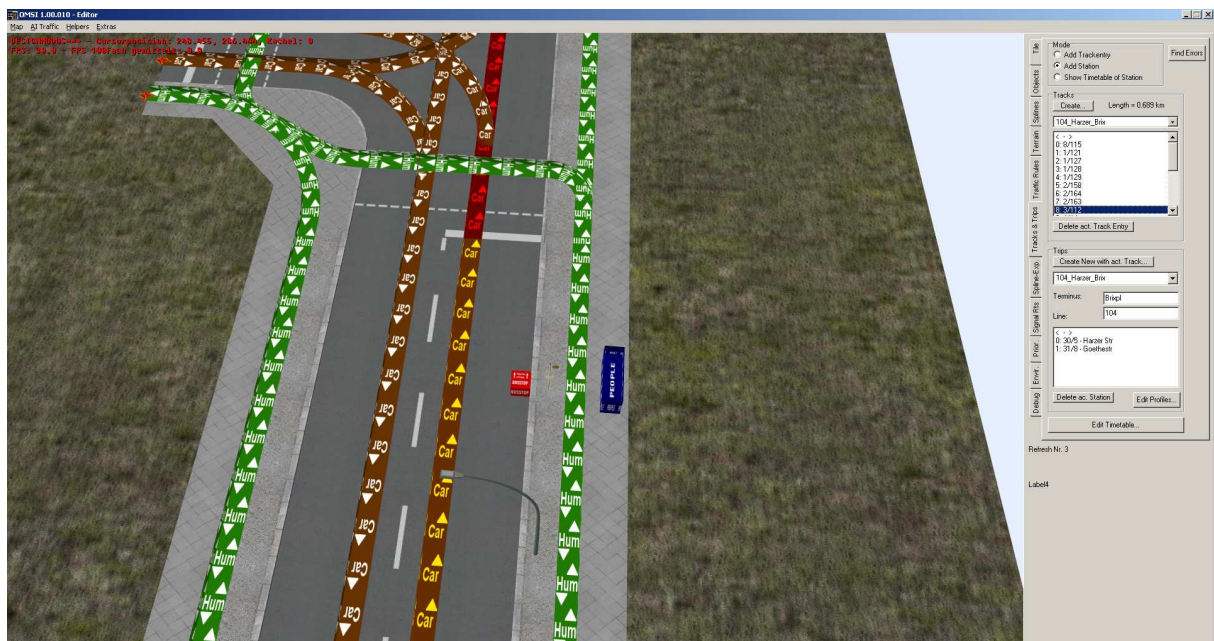
Select the "Add Station" mode at the top. Then, from the list, select the track segment located directly next to the first stop. It will turn yellow:



Now click onto the bus stop cube. *Always* make sure the adjacent track segment is coloured yellow, otherwise there will be errors in the AI traffic! The bus stop will now be displayed in the lower list:



Now select the track segment next to the following stop "Goethestr" and click the bus stop cube:



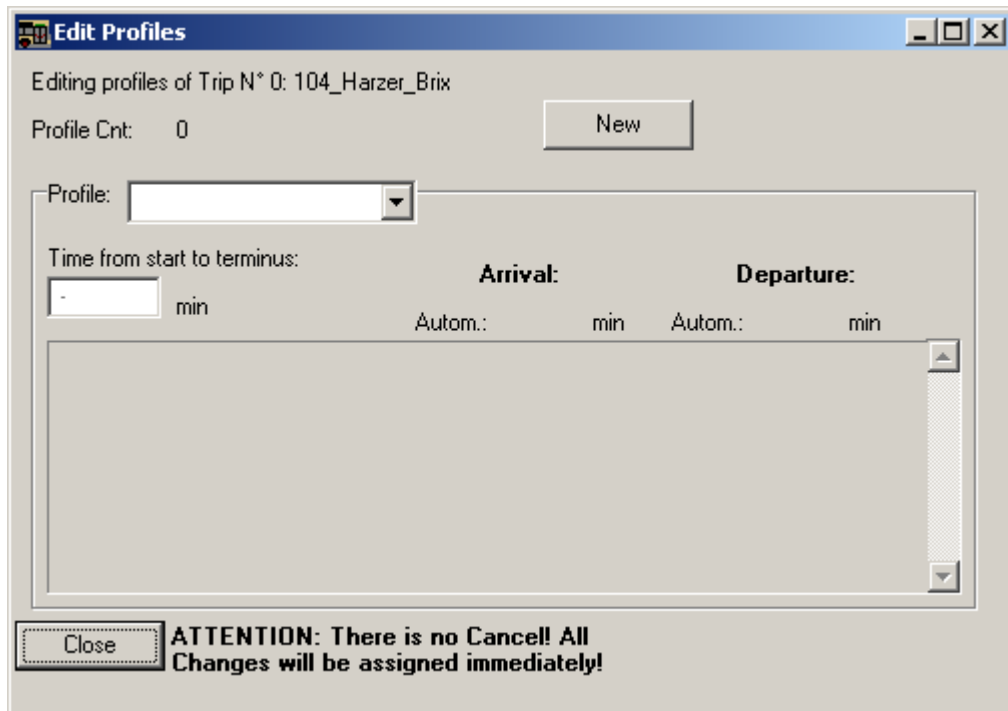
Important: The order of the bus stops will be sorted automatically, derived from the order of the appropriate segments. So you don't need to care about the order and you can later add stops in between.

Now turn to the terminus in the southern loop:



Now you can choose one of the two bus stops. It will then be selected and coloured in red (ok, the cube is already red, but the label will then turn red, too! ☺). Furthermore, the appropriate track segment will be selected as well, so you can check whether the correct track segments are "connected" with their proper bus stops. If it doesn't fit somewhere, just delete the "wrong" bus stop and insert it again!

Next, a profile must be created. Click onto "Edit Profiles...". The following window appears:



Until now, there is no profile. Create a new one by clicking "New". Name it anyway you want (maybe "standard"). The view will change as follows:

Edit Profiles

Editing profiles of Trip N° 0: 104_Harzer_Brix

Profile Cnt: 1 New

Profile: standard

Time from start to terminus: 1.000 min

	Arrival:		Departure:	
	Autom.:	min	Autom.:	min
Harzer Str	<input checked="" type="checkbox"/>	0.121	<input checked="" type="checkbox"/>	0.121
Goethestr	<input checked="" type="checkbox"/>	0.439	<input checked="" type="checkbox"/>	0.439
Brixpl	<input checked="" type="checkbox"/>	0.981	<input checked="" type="checkbox"/>	0.981

Close **ATTENTION: There is no Cancel! All Changes will be assigned immediately!**

Here you can see the time from the starting pint up to the terminus. More exactly: The time from the first to the last track segment. At first, one minute has been assumed. Below, you'll find the list of stations. All checkboxes are checked, to all arrival and departure times will be calculated automatically. Now enter 5 min driving time:

Edit Profiles

Editing profiles of Trip N° 0: 104_Harzer_Brix

Profile Cnt: 1 New

Profile: standard

Time from start to terminus: 5.000 min

	Arrival:		Departure:	
	Autom.:	min	Autom.:	min
Harzer Str	<input checked="" type="checkbox"/>	0.605	<input checked="" type="checkbox"/>	0.605
Goethestr	<input checked="" type="checkbox"/>	2.195	<input checked="" type="checkbox"/>	2.195
Brixpl	<input checked="" type="checkbox"/>	4.906	<input checked="" type="checkbox"/>	4.906

Close **ATTENTION: There is no Cancel! All Changes will be assigned immediately!**

Once you've confirmed by pressing [Enter], the station times will be recalculated.

You can always uncheck a checkbox and enter a manual time. The other times will be adjusted accordingly. In this small example, that's not necessary, the automatically calculated times will do.

Close the window by clicking "Close". The trip is complete!

Now create the opposite trip "104_Brix_Harzer" the same way. Enter "Harzer Str" for terminus and proceed as described above (including the creation of a profile). In this case, there's an anomaly: "Brixplatz" and "Goethestr" are on the same track segment! But that doesn't matter.



After all these preparations, the actual timetable can finally be created! Klick "Edit Timetable..."

Every scheduled trip belongs to a tour. Tours can be merged into lines in turn. The lines are only a means of grouping. You are not forced to create tours that serve only one line. But you can only assign the properties "User is allowed to drive this line" and "Priority" once for a whole line!

Now create a new line and name it "104". You want to drive this line yourself, so click the checkbox „User is allowed to drive this line“. The priority can remain set to 1.

Next you'll need to create a tour. A tour is "one bus" that runs certain trips one after another. In the "Tour" section, click "Create" and name it "1":

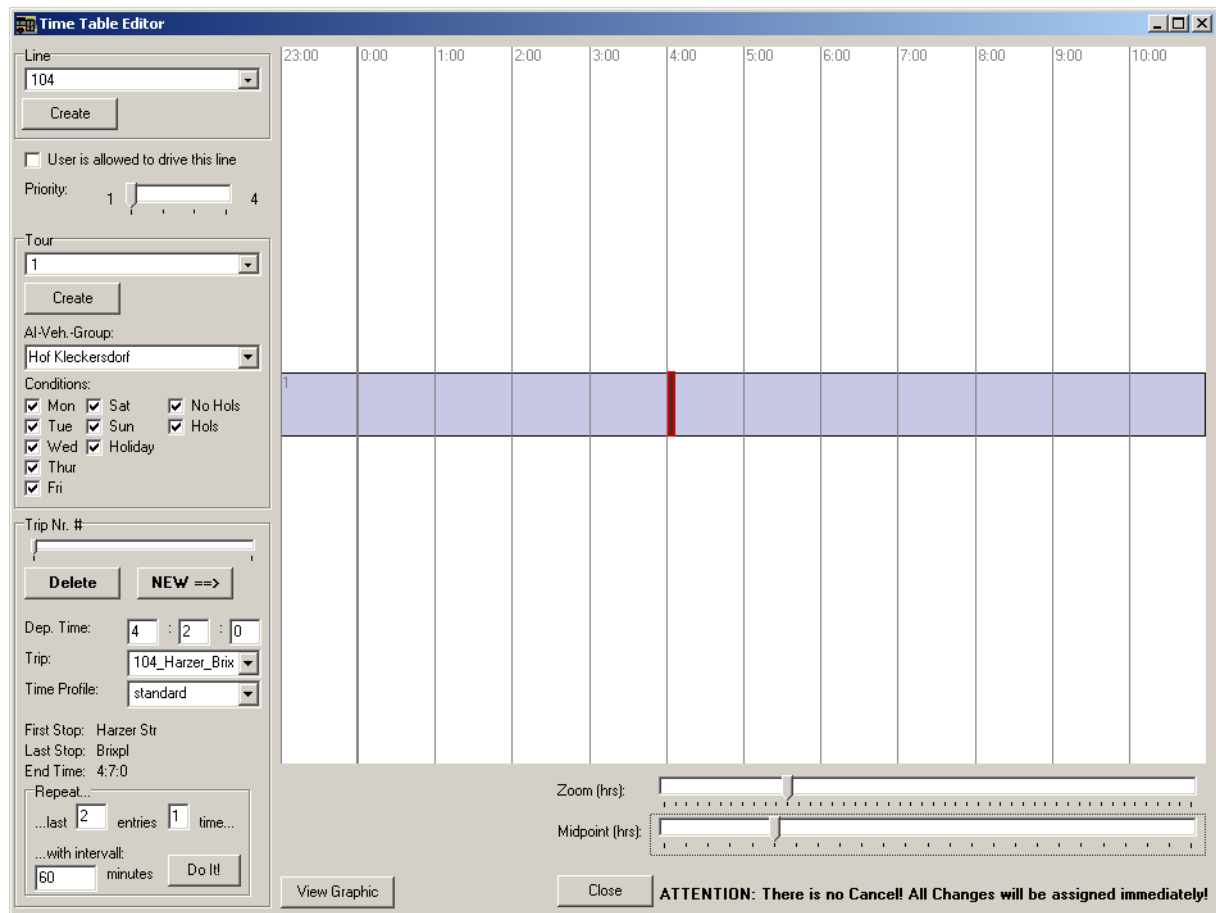
The large chart shows the hours of the day on the x axis and the tours one below the other. For now, there's only one tour, which is still completely empty.

Underneath the "Create" button, you can select the vehicle group you've created earlier, that shall be used on this tour. Thus, you'll chose "Hof Kleckersdorf" ("Podunk depot").

You can assign an individual vehicle group for each tour, thus, you have maximum flexibility when assigning the vehicles: Either all buses are randomly spread on all tours, or every tour belongs to a certain vehicle group, which might contain only one particular bus (we'll get back to that!).

Beneath the vehicle group, you can select the running times for the tour: Only on certain weekdays and/or on public holidays (a holiday doesn't count as a normal weekday). Or, if the tour runs *only* during school holidays (in this case, activate "Hols") or *not* during school holidays (in this case, activate "No Hols") or in both cases (activate both "Hols" and "No Hols").

Beneath, there are controls that can be used to apply individual trips: Klick " New ➔ ". A new trip will be inserted, starting at 00:00. But we want to make the first trip start at 04:02:00, so enter "4:2:0". To examine the graph more closely, you can zoom/move the chart using the sliders underneath the chart:



The chosen trip is "104_Harzer_Brix" - which can remain unchanged. The time profile is "standard" for there is only that particular profile available. Below, the first and the last stop are shown as well as the arrival on the terminus: 4:07.

To add the return trip, click „New ➔“ again and select the trip "104_Brix_Harzer". You can see that the departure time is set to one minute after the preceeding arrival. But we want to insert a break of 5 mins, so the return trip won't start until 4:12:

After 20 mins (including the break), this cycle can start again. For this purpose, there is a very practical function, located at the very bottom: The function allows you to repeat the last x entries y times within the time interval t. By default, these values are set to $x = 2$, $y = 1$ and $t = 60$.

In our case, only the interval must be set to 20 mins. Then click "Do it!":

Now check the entries created so far: Move the slider above „Delete / New ➔” and watch the graph and the entries.

The functions "Delete" and „New ➔” only work if the slider is set to the final position. Thus, it's not possible yet to insert trips in between, which normally isn't necessary anyway. In that particular case, the following entries need to be deleted. But generally, these entries can be easily reconstructed by using the repeat function.

Now zoom out so you can see the whole day:

Time Table Editor

Line: 104
Create

☐ User is allowed to drive this line
Priority: 1 4

Tour: 1
Create

Al-Veh.-Group: Hof Kleckersdorf

Conditions:
☒ Mon ☒ Sat ☒ No Hols
☒ Tue ☒ Sun ☒ Hols
☒ Wed ☒ Holiday
☒ Thur
☒ Fri

Trip No. 1
Delete NEW ==>

Dep. Time: 4 : 12 : 0
Trip: 104_Brix_Harzer
Time Profile: standard

First Stop: Brixpl
Last Stop: Harzer Str
End Time: 4:17:0

Repeat...
...last 2 entries 1 time...
...with interval:
20 minutes Do It!

Zoom (hrs):
Midpoint (hrs):

View Graphic Close

ATTENTION: There is no Cancel! All Changes will be assigned immediately!

Insert further trips using the repeat function:

Important: Departure and arrival times greater than 24:00 are possible! E.g. if a trip doesn't end until past midnight. This principle has been used in Spandau, too. A tour can even last longer than 24 hours! In that case, "yesterday's" bus will meet "today's" bus!

Now you can close the window by clicking "Close".

Save the map and start up OMSI!

If everything has been done correctly, a D92 AI bus should be driving around!



Unfortunately, this bus doesn't care about bus stop signs and passengers:



You can correct that by moving the bus stop cube a bit further towards the center of the road!

Basically, the scheduled traffic should work quite well now. Furthermore, you should be able to start a scheduled trip with your own bus.

5.4. *Adding your own bus repaint*

Though it's not exactly part of *this* manual, we will now explain how to create a repaint. The reason is that we want to demonstrate afterwards

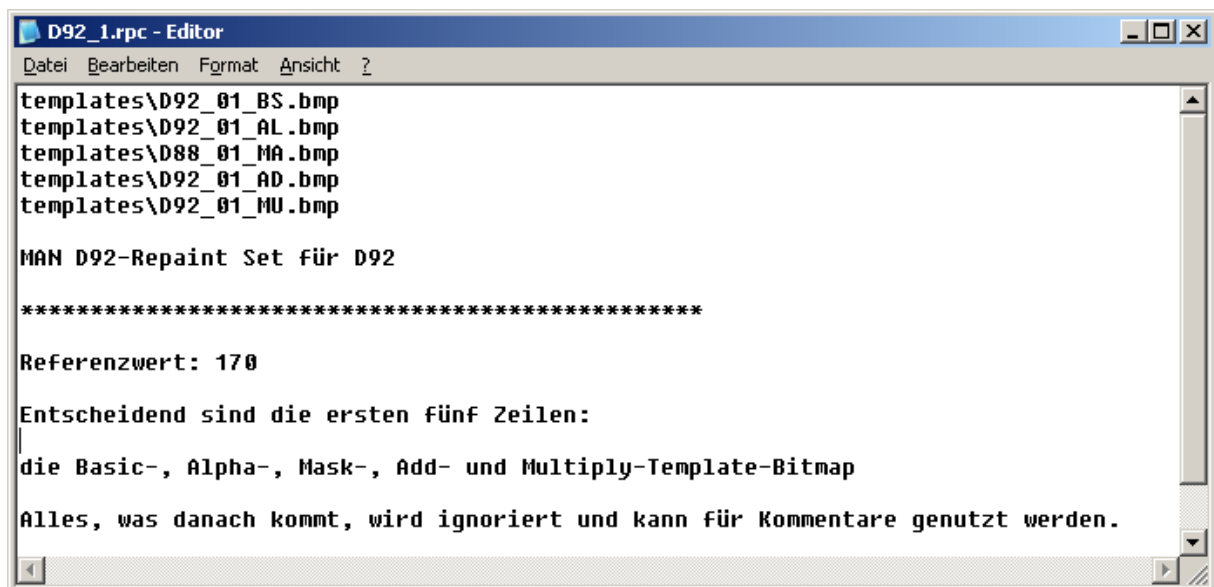
how to set up an AI vehicle list containing only certain numbers and liveries.

It is fery easy to create a repaint, thanks to the Repaint-Tool which is part of the SDK.

Both doubledeckers SD200 and SD202 have two outside textures each, "_1" and "_2". "_1" contains sides, front and back of the bus, the roof is part of "_2".

If you (as in this case) only want to add a paint job to the side of the bus, editing the "_1" texture is sufficient.

Change to the SDK directory and open "Repaint_Tool \ SD202". There you'll find a set of repainter files. Open the file D92_1.rpc using the Windows Editor.



This file is used by the repainter to determine which files in the template folder will be used to repaint a D92, if the livery is restricted to the "_1" texture (front, back and sides).

Too academic? Let's put it into practice!

Copy the file „D92_01_BS.bmp“ from the template folder into the „Repaint_Tool\SD202“ folder and rename it to „D92_01_Testbus.bmp“! You can then open the file, e.g. with Paint:



This file will be used as background (Base, BS). The other files mentioned in the rpc file are used to create additive and multiplicative overlays.

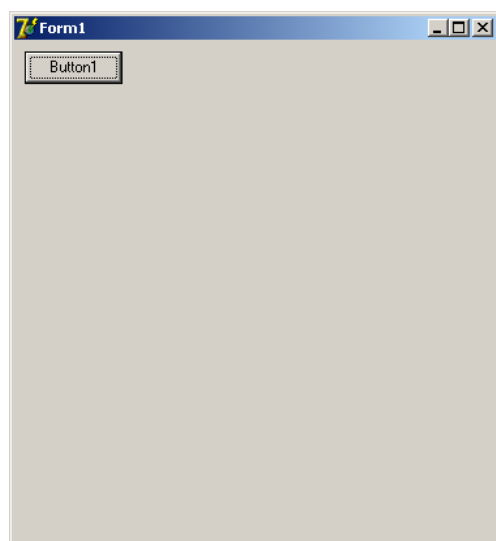
Use this texture to paint your first test livery! Areas that shall not be "pasted", need to be painted with pure magenta:



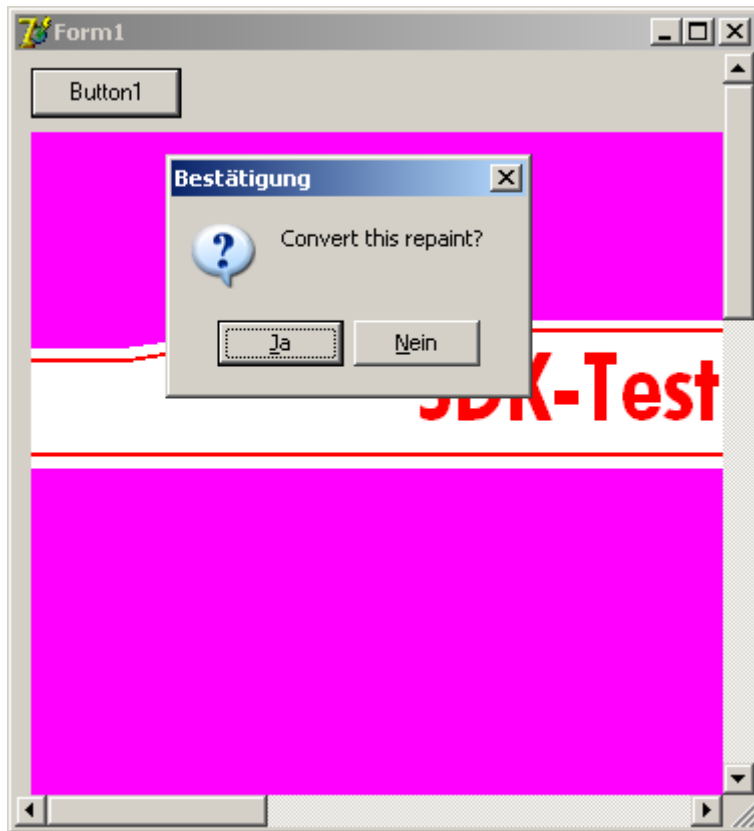
When finished, there should be nothing left of the original texture:



Now start "RepaintTool.exe":



Click "Button1" and first choose the appropriate rpc file for your livery (see appendix for the rpc files used by the other SD202 and SD200!): „D92_1.rpc“. After that, choose "D92_01_Testbus.bmp":

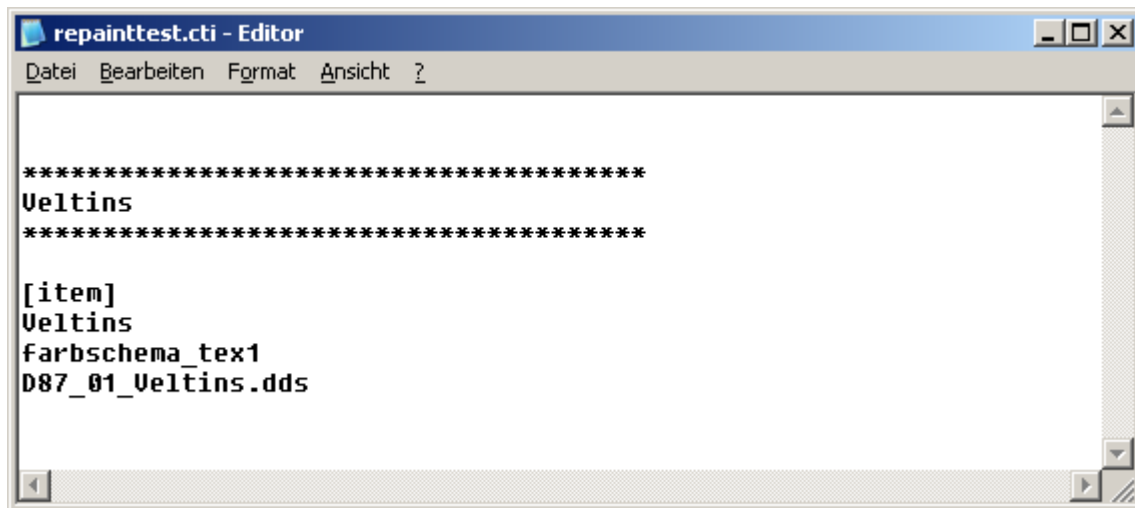


Confirm by clicking "Ja". The repaint tool will now create the file „D92_01_Testbus.dds“.

Have a look into the directory „Omsi \ Vehicles \ MAN_SD202 \ Texture“! There you'll find a folder called "Werbung" (advertisement) for each SD202 series. Now copy the file "werbungen1_zugelassen.cti" from the folder "Werbung_D87" into the folder "Werbung_D92" and rename the file to "repainttest.cti".

Important: This file contains all necessary information about one or more OMSI repaints. OMSI always searches for all cti files in a particular folder, i.e. you can (and should) always create your own cti file for your own repaints *without editing* the existing cti files.

Now open the file using Notepad:

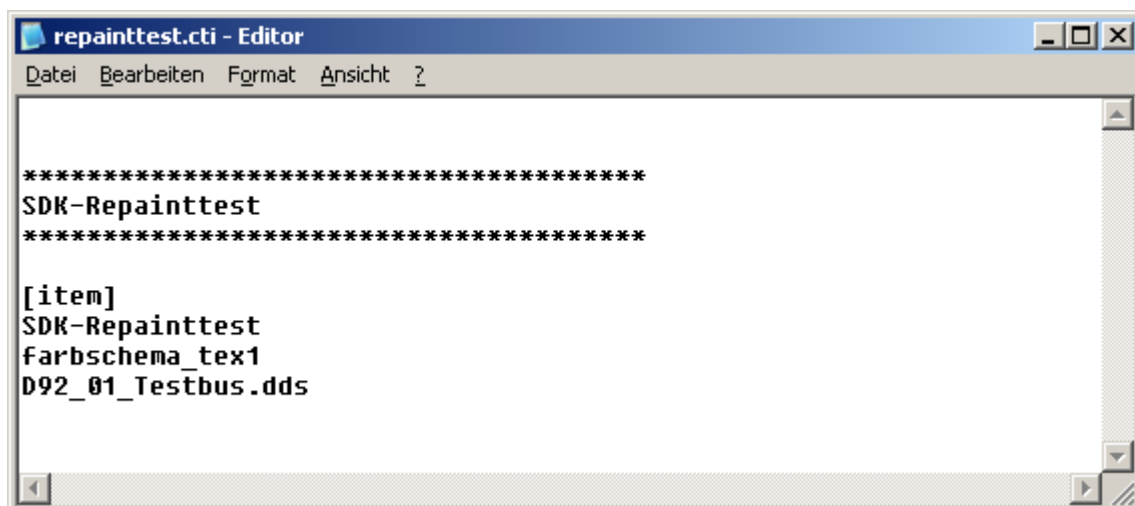


With this file being another configuration file, the "header" " (** Veltins **) will be interpreted as a commentary. The real information begins with [item]:

- Veltins = Name (for ailists.cfg and the vehicle selection dialogue)
- farbschema_tex1 = internal identifier so that OMSI knows how to use this texture
- D87_01_Veltins.dds = change texture

There will always be one [item] entry per livery *and* per texture. To create a livery including the roof, you will need two entries.

The file now needs to be adjusted as follows:



Furthermore, the „D92_01_Testbus.dds“ texture must be copied into the "Werbung_D92" directory.

Then, the repaint should already be available in the vehicle selection dialogue:



If you want to extend this repaint over the roof, you will need to create a roof texture using "D90_02_BS.bmp" (which is also valid for the D92, see appendix). This texture should then be called „D92_02_Testbus.bmp“ consistently. After the conversion ("D92_2.rpc"), this file will be named "D92_02_Testbus.dds".

Then you'll need to create a second [item] entry:

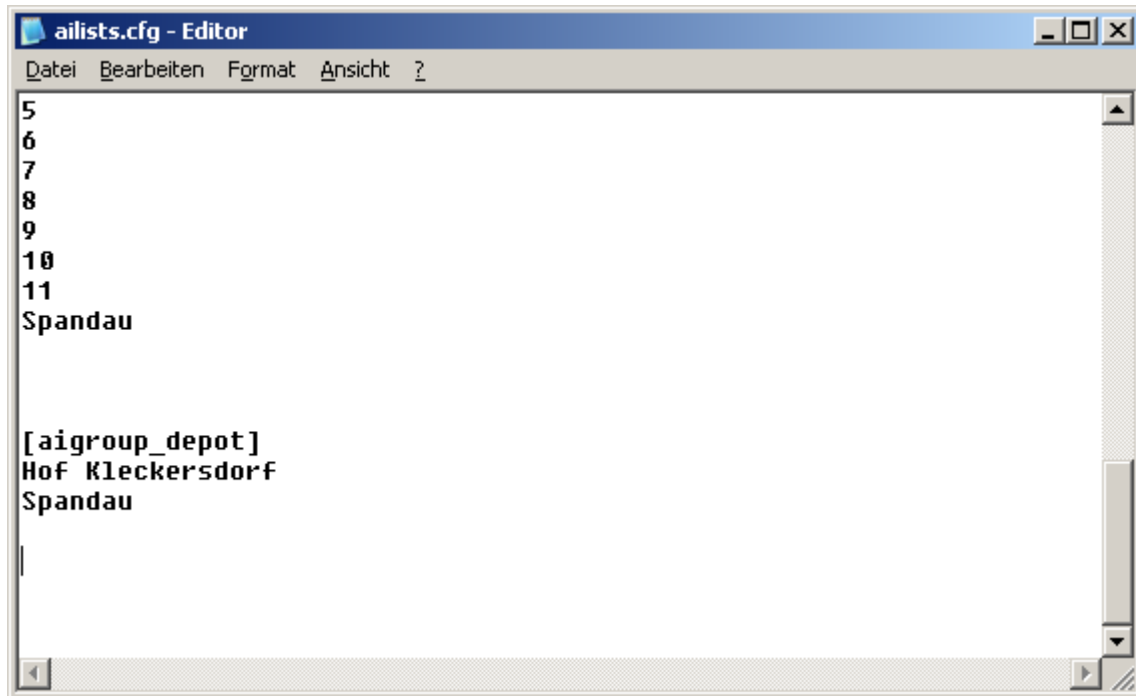
```
[item]
SDK-Repainttest
farbschema_tex2
D92_02_Testbus.dds
```

5.5. *AI groups with specific numbers*

We have already worked with AI groups. Now we want to change the existing AI group in such way that we can specifically define bus numbers and liveries of that group.

Open the ailst.cfg in the map directory of "Podunk" ("Klein Kleckersdorf").

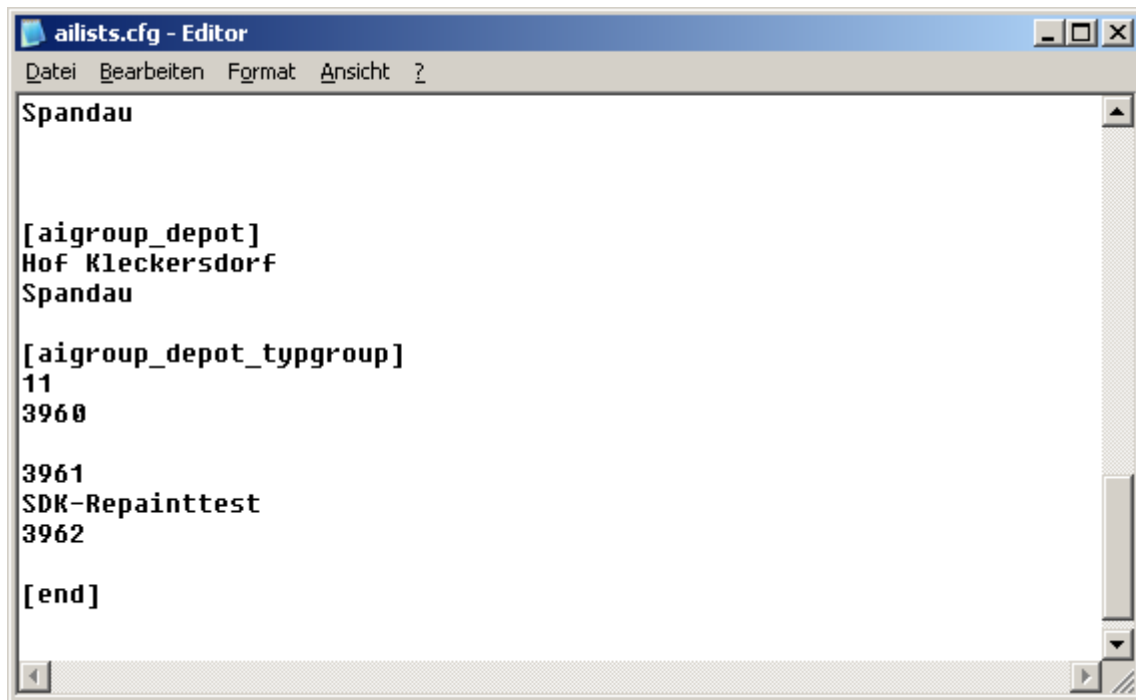
Add a "_depot" to the [aigroup] "Hof Kleckersdorf" ("Podunk depot") and delete the two numbers and 11:



OMSI now expects a list of vehicle numbers to be inserted!

This looks as follows: It begins with the entry „[aigroup_depot_typgroup]“. After that, there comes the vehicle type index (11 again for "MAN_D92.bus").

After that, any number of pairs consisting of "bus number - livery" can be entered and at the very end, an [end] mark must be placed to terminate the list:



This means that here are three buses in the group: Bus 3960 without advertisement, bus 3961 with the repaint test livery created just now and the bus 3962 without advertisement.

Important: The spelling of the advertisements (apart from the case in which there's none) must exactly match the spelling in the cti file (or in the vehicle selection dialogue)!

If you start OMSI now (after saving), you will notice that always one of the three defined buses will serve your line!



You can add as many lists of „[aigroup_depot_typgroup]“ as you like. For instance, you can create a second list with two SD85 showing the LFH advertisement:

```
ailists.cfg - Editor
Datei Bearbeiten Format Ansicht ?

[aigroup_depot]
Hof Kleckersdorf
Spandau

[aigroup_depot_typgroup]
11
3960

3961
SDK-Repainttest
3962

[end]

[aigroup_depot_typgroup]
7
3413
LFH (Pop)
3434
LFH (Pop)
[end]
```



5.6. Creating a depot file

Finally, we will explain how to create a depot file for the SD202.

Change to the "MAN_SD202" directory, copy the file "Grundorf.hof" and rename it to "Podunk.hof".

Open the file with Notepad

Don't get confused by the mass of commentary. As usual, only the keywords matter.

At first, there is the keyword [name], followed by the name. Change the name to "Podunk"!

The next entry [servicetrip] is necessary so that OMSI knows what to display on a service run (e.g. when the bus is put out of service). This entry can remain unchanged.

Next comes the keyword "stringcount_terminus" (this time without brackets!), followed by the number of strings associated with each terminus display. Beneath, as commentary, you find the explanation:

String_0: IBIS-Display (max. 16 chars, upper case only)

String_1: Matrix display, front, 1st line (max. 16 chars, upper case only)

String_2: Matrix display, front, 2nd line (max. 16 chars, upper case only)

String_3: Matrix display, side, for SD200 (max. 16 chars, upper case only)

String_4: rollsign texture (only used by SD200, line must be allocated anyway!)

String_5: IBIS2-Display (only D92! 20 chars, upper and lower case possible)

After that, you will find the keyword "stringcount_busstop" with the number of text strings (4) per bus stop. The coding is as follows:

String_0: IBIS-Display (max. 16 chars, upper case only)

String_1: interior bus stop display, 1st line (upper and lower case)

String_2: interior bus stop display, 2nd line (if the display shall change between 1st and 2nd line, otherwise leave the line empty „")

String_3: IBIS2-Display (max. 20 chars, upper and lower case)

The list of all terminus entries will be next. The order is important for the buses fitted with rollsigns: The terminus displays will be inserted into the rollsign in the order in which they are listed here!

There are two types of entries: "[addterminus]" for normal terminus displays and "[addterminus_allexit]" for displays that always tell the passengers to get off/not to get on. (e.g. "Betriebsfahrt", service run).

The keyword is followed by the destination code for the IBIS, the appropriate bus stop (as spelled on the red bus stop cube and in the appropriate trips) and the number of strings defined at the beginning:

Example:

[addterminus]	Schlüsselwort
---------------	---------------

102	IBIS-code number
Saganoallee	appropriate bus stop
SAGANOALLEE	IBIS-isplay (IBIS 1)
	first line front matrix
SAGANOALLEE	second line front matrix
SAGANOALLEE	side matrix for SD200
Gru_Saganoallee.tga	texture name for sollring texture (SD200)
Saganoallee	IBIS-2-display
.....	only 16 non-functional dots to show the maximum width for matrix and IBIS displays

If you want to create a deput file for rollsign SD200:

The appropriate rollsign textures are located in the folder „Vehicles \ Anzeigen \ Rollband_SD79“. You can add your own textures there.

These textures must be stored in a format using the alpha channel (e.g. tga). You must not set the entire texture to an alpha value of 255, at least one pixel must have an alpha value of 254 or less, otherwise there will be graphics errors!

After the list of terminuses, the regular bus stops are listed. They consist of the following entries:

[addbusstop]	keyword
Einsteindorf Ausbau	appropriate bus stop
D.DORF AUSBAU	IBIS-display (IBIS 1)
Einsteindorf	interiour bus stop display, 1st line
Ausbau	interiour bus stop display, 1nd line (changeover)
Einsteindorf Ausbau	IBIS-2
.....	dotted line as reference for 20 chars
.....	dotted line as a reference for 16 chars

At last, you'll find the routes. They olways consist of two parts: An "[infosystem_trip]" antry and an "[infosystem_busstop_list] entry:

[infosystem_trip]	keyword
7602	Line + Route (Line 76, Route 02)
KRANKENHAUS-BAUERNHOF	(not in use)
107	appropriate terminus code
76	(not in use)

The "[infosystem_busstop_list]" entry on the other hand only consists of the number of the following entries and the entries themselves. As before, the bus stop names must match the names entered on the red bus stop cubes that have already been used in the previous [addbusstop] entries.

A detailed tutorial explaining how to insert your own destination displays, bus stops and routes would lead too far. We hope that by now, your experience is sufficient to try it yourself. If you have any queries, please do not hesitate to contact us on our forum!

Appendix 1: Intersections and matching spline types

Intersection:	Crossing splines:
bue_falks_ohe.sco	Bahnübergang über str_6spur_falkenseer1.sli
Einm_Altonaer.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_Blasew_Obst_Sand.sco	str_2spur_12m_Sandstr.sli oder str_2spur_12m_SeeburgerWeg1.sli
Einm_Blasew_Reclam.sco	str_2spur_12m_Sandstr.sli mit str_2spur_9m_Reclamweg1.sli
Einm_Borkumer.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_Brunb_Whv.sco	str_2spur_10m_brunsbuetteler1.sli mit str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_erzgebirgs.sco	str_2spur_6m_erzgebirgs.sli (oder str_2spur_6m_freud1.sli)
Einm_falks_remscheider.sco	str_6spur_falkenseer1.sli mit str_2spur_6m_erzgebirgs.sli
Einm_falks_steigerwald.sco	str_6spur_falkenseer1.sli mit str_2spur_6m_erzgebirgs.sli
Einm_frankenwald_erzgebirgs.sco	str_2spur_9m_westerwald1.sli oder str_2spur_9m_frankenwald1.sli mit str_2spur_6m_erzgebirgs.sli
Einm_freud_kraepelin.sco	str_2spur_6m_freud1.sli
Einm_Sedan.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_See.sco	str_2spur_11m_SeeburgerStr1.sli
Einm_see_johanna.sco	str_2spur_11m_SeeburgerStr1.sli mit str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_See_Paewesiner.sco	str_2spur_11m_SeeburgerStr1.sli mit str_2spur_8m_altonaer1.sli,

	str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_see_seeckt.sco	str_2spur_11m_SeeburgerStr1.sli mit str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Einm_westerwald_erzgebirgs.sco	str_2spur_9m_westerwald1.sli oder str_2spur_9m_frankenwald1.sli mit str_2spur_6m_erzgebirgs.sli
Extra_002_Wende1.sco	str_2spur_6m_freud1.sli (oder str_2spur_6m_erzgebirgs.sli)
Kreuz_Altonaer.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Kreuz_Borkumer.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Kreuz_erzgebirgs.sco	str_2spur_6m_erzgebirgs.sli (oder str_2spur_6m_freud1.sli)
Kreuz_falks_frankenwald.sco	str_6spur_falkenseer1.sli mit str_2spur_6m_erzgebirgs.sli (oder str_2spur_6m_freud1.sli) und str_2spur_9m_westerwald1.sli oder str_2spur_9m_frankenwald1.sli
Kreuz_falks_kiesteich.sco	str_6spur_falkenseer1.sli mit str_2spur_9m_westerwald1.sli oder str_2spur_9m_frankenwald1.sli und str_2spur_12m_kiesteich1.sli
Kreuz_falks_westerwald.sco	str_6spur_falkenseer1.sli mit str_2spur_9m_westerwald1.sli oder str_2spur_9m_frankenwald1.sli
Kreuz_Sedan.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Kreuz_See_Elsflether.sco	str_2spur_11m_SeeburgerStr1.sli mit str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Kreuz_see_schmknob_lutoner.sco	str_2spur_11m_SeeburgerStr1.sli mit str_2spur_12m_SeeburgerWeg1.sli
Kreuz_westerwald_erzgebirgs.sco	str_2spur_9m_westerwald1.sli oder str_2spur_9m_frankenwald1.sli mit str_2spur_6m_erzgebirgs.sli

Verzieh_falks_zepp1.sco	str_6spur_falkenseer1.sli
Verzieh_falks_zepp2.sco	str_6spur_falkenseer1.sli
Wende_Borkumer_R.sco	str_2spur_8m_altonaer1.sli, str_2spur_8m_borkumer1.sli oder str_2spur_8m_sedan1.sli
Wende_Erzgebirgs_L.sco	str_2spur_6m_erzgebirgs.sli (oder str_2spur_6m_freud1.sli)
Wende_Erzgebirgs_R.sco	str_2spur_6m_erzgebirgs.sli (oder str_2spur_6m_freud1.sli)

Appendix 2: Repainter files for SD200 and SD202

SD200:

	Front/side/back	Roof
SD77	SD77_01.rpc	SD77_02.rpc
SD80	SD80_01.rpc	SD80_02.rpc
SD81	SD81_01.rpc	SD81-82_02.rpc
SD82	SD82_01.rpc	
SD83	SD83_01.rpc	SD83_02.rpc
SD83_RLB	SD83_RLB_01.rpc	
SD84	SD84_01.rpc	SD85_02.rpc
SD85	SD85_01.rpc	

SD202:

	Front/side/back	Roof
D86	D86_1.rpc	D86_2.rpc
D87	D87_1.rpc	
D88	D88_1.rpc	D88_2.rpc
D89	D89_1.rpc	
D92	D92_1.rpc	D92_2.rpc

