

The Big Book of SnowRunner Map Making

Revision 0.1 (2021-08-07)

Copyright 2021 Chris Nelson



This work is licensed under a [Creative Commons Attribution-NonCommercial 4.0 International License](#).

If you own the Windows version of SnowRunner, then you can make and publish your own custom maps.

This guide documents every feature of the SnowRunner map editor, plus how to create custom objects for your map. You can read (or skim) sequentially through the guide to learn everything there is to know, or you can use it as a reference when you want to learn more about a particular feature.

What I put in this guide that I can't find elsewhere:

- Information about **all** editable map properties.
- Full details for complex editing tasks.
- Bugs in the editor and the game to avoid.
- How to create and use reference maps.
- How to create custom materials, distribution brushes, roads, dynamic models, and even plants.
- Easy reference material for trucks, trailers, addons, and their permitted combinations.
- Tips for using Blender to create custom objects for SnowRunner.
- And more!

Table of Contents

Quick Start.....	5	Edit a Property Using a Slider.....	42
Run the Map Editor.....	5	Edit a Property That Selects an Asset.....	43
Create a Fresh Map.....	6	Edit a Property that Selects Brushes.....	45
Move the Camera.....	8		
Make Hills and Depressions.....	10	The Context Menu.....	47
Place a Truck.....	13	Move a Truck, Model, or Plant.....	53
Save and Pack.....	17	Manipulate an Object in the Main Panel.....	53
Make a Backup Copy of your Campaign.....	18	Manipulate an Object in the Scene View.....	54
Test Your Map.....	18	Land and Do Land.....	55
Re-editing a Map.....	21	Local Transform.....	58
Example Maps.....	21	Edit Other Truck Properties.....	59
How to Use This Book.....	22	Select Truck Components.....	59
Sources of Information.....	22	Choose the Truck Type.....	59
Is This Book Up To Date?.....	22	Identify Compatible Truck Parts in the Virtual	
Editor and Game Bugs.....	23	Garage.....	60
Design Tips.....	23	Choose an Engine, Gearbox, Winch, and	
If You Made Maps in MudRunner.....	24	Suspension.....	60
Directories, Files, and Archives.....	25	Choose Wheels.....	61
Prebuild Directory.....	25	Choose a Color.....	61
Create an Archive.....	26	Choose a Trailer.....	62
Levels Directory.....	26	Choose Add-ons.....	62
Mods Directory.....	27	Truck ID.....	64
AppData Directory.....	27	Starting Damage and Fuel.....	65
Config Directory.....	28	Locked and Unlocked Vehicles.....	65
The Map Editor Window.....	29	Reserved Trucks.....	66
Menu Bar.....	29	The Active Truck.....	66
Toolbar.....	31	Hand Edit the XML.....	67
Main Panel.....	33	Terrain Height.....	68
Navigate in the Main Panel.....	33	Height Brush Mode.....	68
Select Targets in the Main Panel.....	34	Autofade.....	70
Other Actions in the Main Panel.....	34	Flatten Brush Mode.....	71
Output Panel.....	35	Smooth Brush Mode.....	72
Terrain Panel.....	35	Brush Keyboard Shortcuts.....	73
Navigate in the Terrain Panel.....	36	Height Information.....	73
Select Targets in the Terrain Panel.....	36	Deceptive Height Cues.....	74
File View Tab.....	36	Navigate While Painting.....	75
Controls for Hierarchical Lists.....	37	Restore a Previous Terrain State.....	76
Scene View Panel.....	38	Mud.....	77
Select Targets in the Scene View Panel.....	38	Extrudes Mode.....	77
Introduction to Features.....	39	Extrudes to Wetness.....	80
Property Panel.....	39	Automatic Mode.....	81
Edit a Property by Typing Directly.....	41	Automatic Mode With Size > 2.5.....	81
Edit a Property Using a Dropdown Menu.....	41	Automatic Mode With Size \leq 2.5.....	82
		Decrease Automatic Mud.....	82

Restore a Previous Mud State.....	83
QuickMud.....	84
Materials.....	85
Material Properties.....	86
Texture Layer Properties.....	87
Paint a Texture Layer.....	88
Value Blending.....	90
Edge Blending.....	91
Bleed of 3-D Objects.....	91
Additional Materials.....	92
Texture Edges Between Materials.....	95
Snow Layers.....	97
Delete a Material.....	99
Snow Depth.....	100
Update Texture Layer.....	101
Zones.....	102
ID.....	102
Name and Icons.....	102
Zone Perimeter.....	104
Ribbon Shape and Size.....	105
Vertical Ribbon.....	105
Flat Ribbon.....	106
Other Ribbon Properties.....	108
Hilly Zones.....	108
Zone Settings Editor.....	109
Manage the Zone ID.....	110
Add or Delete a Zone.....	110
Rename a Zone ID in the Main Editor.....	110
Rename a Zone ID in the Zone Settings Editor.....	111
Clear the Zone ID.....	111
Uncommitted Id Edits.....	112
Duplicate IDs.....	112
Zones Used by Objectives.....	113
Zone Settings Editor Controls.....	113
Zone Types.....	115
Garage Entrance.....	115
Garage Exit.....	115
Recovery Zone.....	116
Trailer Store.....	117
Fuel Station.....	117
Service Hub.....	117
Watchpoint.....	117
Free Upgrades.....	119
Automatic Cargo Loading Zone.....	120
Manual Cargo Loading Zone.....	122
Crafting Zone.....	123
Multiple Zone Types.....	123
Objectives.....	124
Objective Status Terminology.....	124
Add an Objective.....	125
UI Text.....	125
Locked Objectives.....	127
Initial Contracts.....	127
Offer Zone.....	127
Global or Regular Zone ID.....	128
Objective Overview.....	129
Stages.....	130
Cargo Delivery Goal.....	131
Spawn Cargo for Stage.....	137
Rewards.....	141
Completion.....	141
Models.....	143
Individual Plants.....	144
Distributions.....	145
Localization.....	146
Playing Your Map.....	148
Dev Tools.....	148
Dev Tools: Create.....	148
Dev Tools: Reload.....	149
Dev Tools: Information.....	149
Dev Tools: Virtual Garage.....	149
Dev Tools: Free Camera.....	150
Dev Tools: Cargoes.....	151
Dev Tools: Repair & Refuel.....	151
Dev Tools: Change Time.....	151
Dev Tools: Mod Tools.....	151
Revealing the Map.....	151
Discovery of Vehicles.....	152
Appendix A: Reference Information	
Truck Reference.....	155
Table Column Meanings.....	155
Scout Trucks.....	157
Highway Trucks.....	157
Heavy Duty Trucks.....	157
Offroad Trucks.....	158
Heavy Trucks.....	159
Texture Layer Reference.....	160
Grass.....	160
Dirt.....	160
Sand.....	161

Mud.....	161
Gravel.....	161
Rocks.....	161
Stone.....	161
Concrete.....	162
Asphalt.....	162
Ice.....	162
Snow.....	162
Zone Icons.....	163
Travel Services.....	163
Objectives.....	164
Cargo Areas.....	165
Vehicles.....	166
Ruins.....	167
Miscellaneous.....	168
Additional Icons.....	168
Cargo Types.....	169
Pallets.....	169
Crates and Containers.....	169
Frames and Straps.....	170
Vehicle Sections.....	170
Logs.....	171
Appendix B: Hand Edit the Map Files	
Material Layout Bitmap.....	172
Fix the Material Layout Alpha Channel in GIMP	172

Quick Start

Before getting into the nitty gritty of map editing, you'll want to create and try your first simple map as quickly as possible.

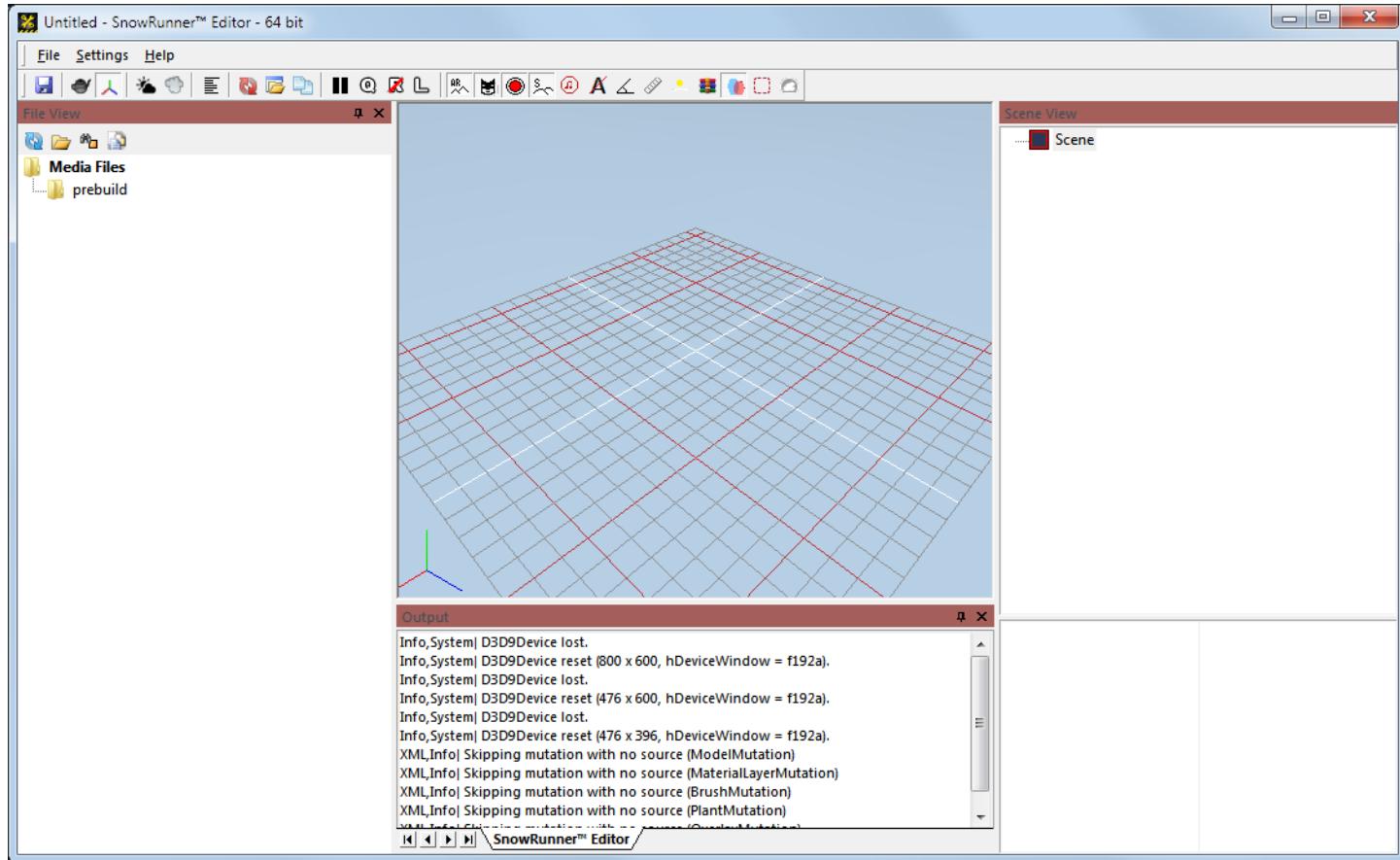
Run the Map Editor

Installing the SnowRunner game also automatically installs the editor, but it doesn't install a convenient shortcut icon. You can start the editor from a path that looks something like this, depending on whether you have the Epic or Steam version of the game:

```
C:\Program Files\Epic Games\SnowRunner\en_us\Sources\BinEditor\SnowRunnerEditor.exe  
C:\Program Files (x86)\Steam\SteamApps\common\SnowRunner\Sources\BinEditor\SnowRunnerEditor.exe
```

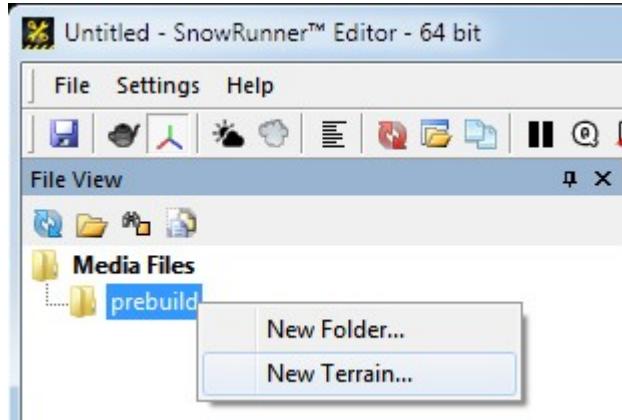
Saber claims that you can't run the SnowRunner Editor and the SnowRunner game at the same time, but I find that they run just fine simultaneously. But if you have any trouble, exit the game before starting the editor.

The initial map editor window looks like this:



Create a Fresh Map

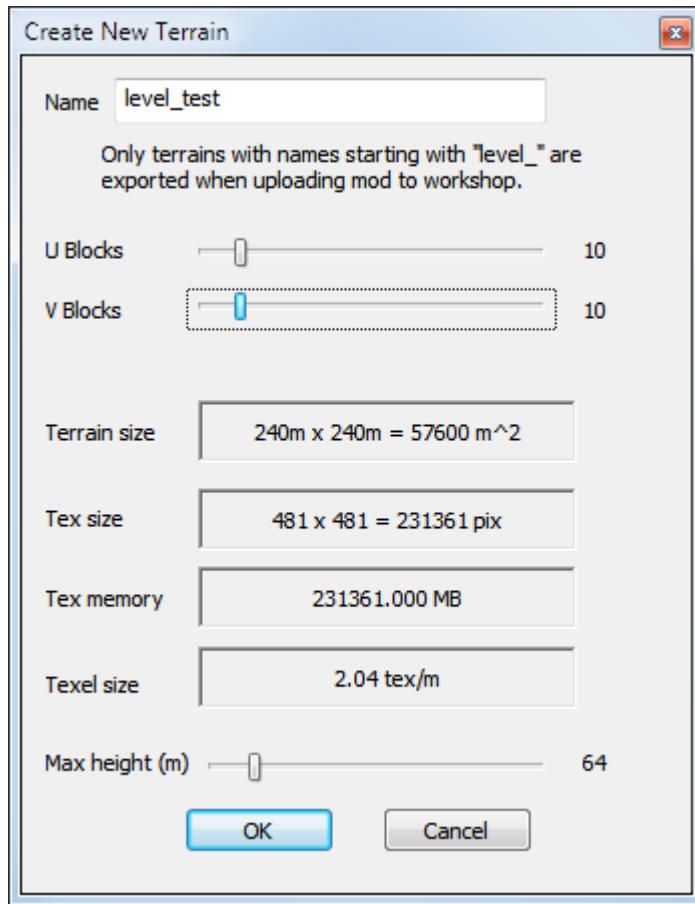
To create a new map, right-click on `prebuild` under `Media Files`, then select `New Terrain...`



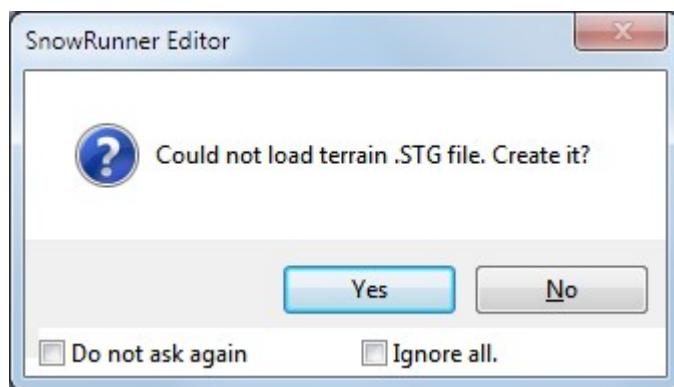
A dialog box pops up asking what size to make the map. The map name **must** begin with `level_`, so in this example I'll call it `level_test`. The default is a rather small map 64 meters on a side, so you might want to crank up the sliders for `U Blocks` and `V Blocks` to something like 10 and 10, which equates to 240 meters on a side. The default max height of 64 meters is fine.

Don't go wild with either of these values. A map with large U and V dimensions will be very slow to work with, and a large max height causes stair-stepping of the terrain.

When you're ready, click `OK`.

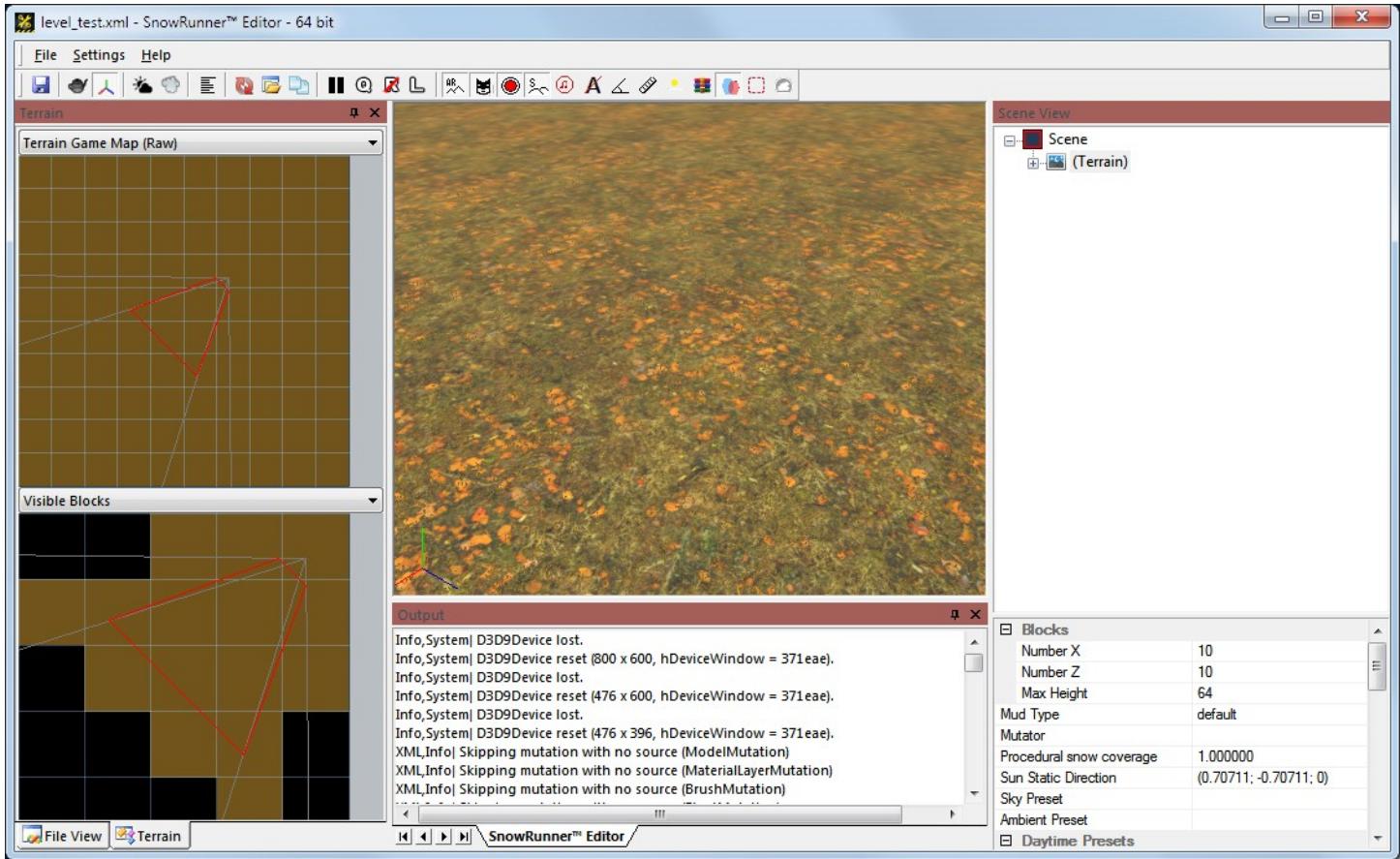


Because this is a new map, it does not yet have a terrain `.STG` file. Click `Yes` to create one.



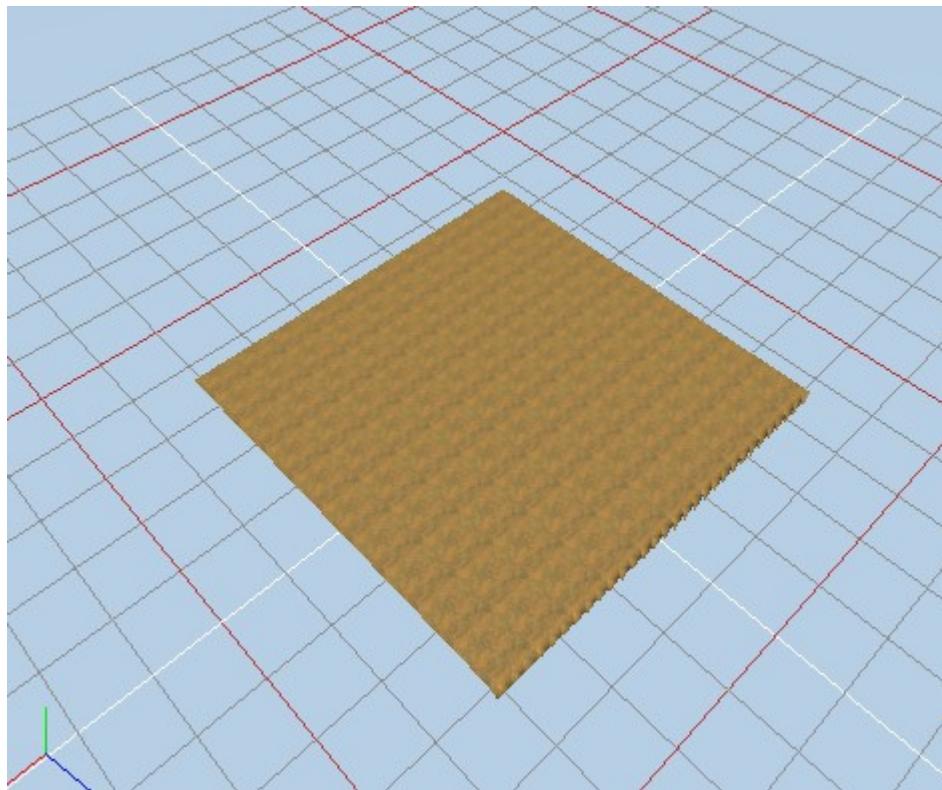
On the right side of the window, the `Scene` updates to include `(Terrain)`.

Bug: Your first time running the editor, the camera is pointed in completely the wrong direction to see your terrain. To point the camera in the correct direction, double-click in the `Terrain Game Map` panel on the left. The main (center) panel now shows you a close-up look at the ground.



Move the Camera

Click in the main panel, then rotate the mouse scroll wheel up and down to zoom in and out. When you zoom out far enough, you can see the edges of the map.



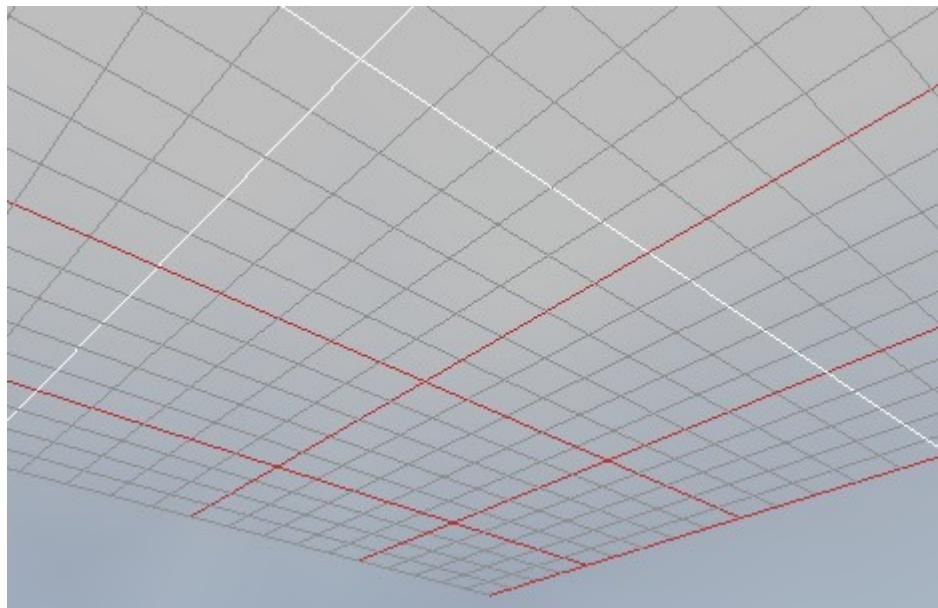
When you zoom in, the camera moves toward where the mouse is pointing. So an easy way to move the camera around the map is to zoom out, point the mouse at the area of interest, and zoom back in.

If you lose track of your terrain, double-click in the **Terrain Game Map (Raw)** panel on the left to restore the camera to a reasonable location.

You can point the camera in different directions by holding down the left mouse button in the main panel and moving the mouse. The camera pivots around the point that you were pointing at when you clicked the button, moving left, right, up, and down.

It is possible to move the camera below the ground. You're not expected to look at the terrain from underneath, so the terrain is only drawn from the top side. So when the camera is underground, all you see is the reference grid above. Rotate the camera back above the ground to restore your vision.

When the camera is pointed up, the background is shaded cloud gray. When the camera is pointed down as is usual, the background is shaded a pale blue.

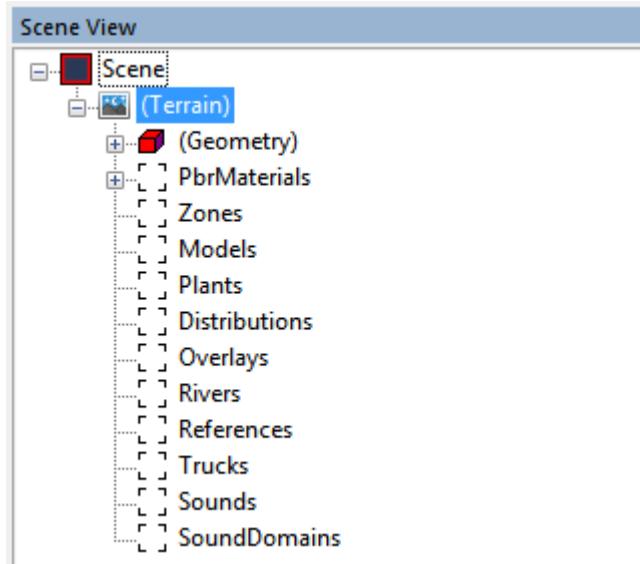


Bug: While moving the mouse to rotate the camera, the mouse pointer also moves. If it moves outside of the main panel, camera rotation will stop.

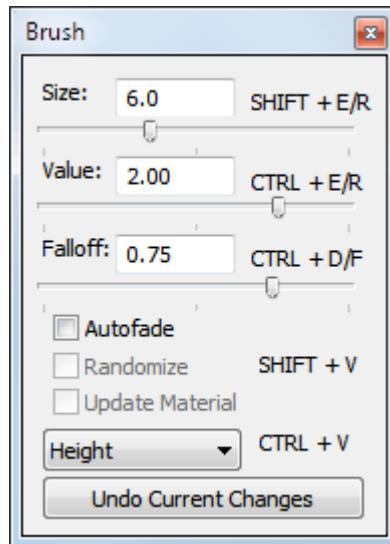
Make Hills and Depressions

The initial terrain is perfectly flat, which isn't very interesting. You can raise the terrain in some places and lower it in other places to make a more interesting terrain to drive around in.

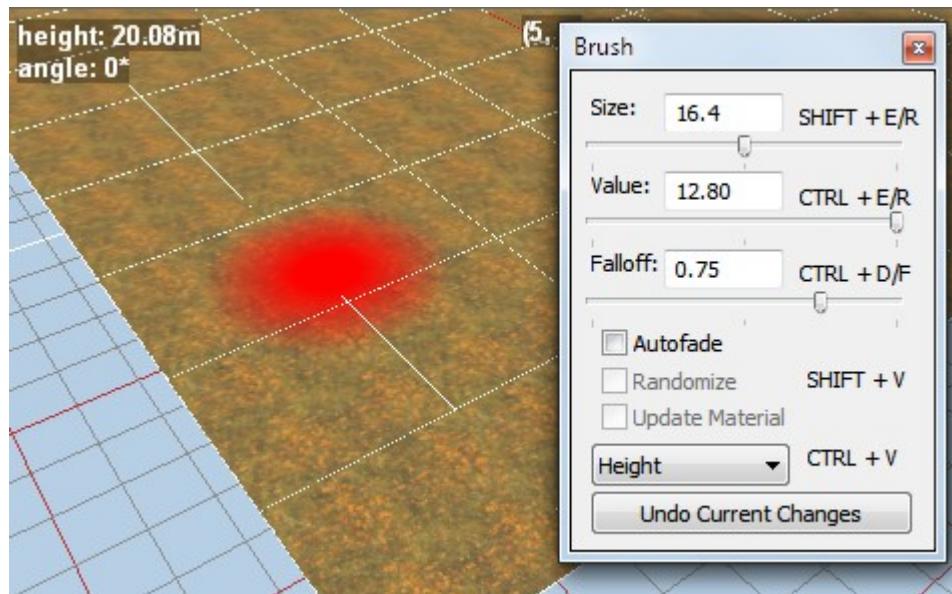
To edit the terrain height, first expand the hierarchy under [Scene→\(Terrain\)](#) to show the many editable aspects of your map, including its [\(Geometry\)](#). There is additional hierarchy within [\(Geometry\)](#), but you don't need to look at that yet.



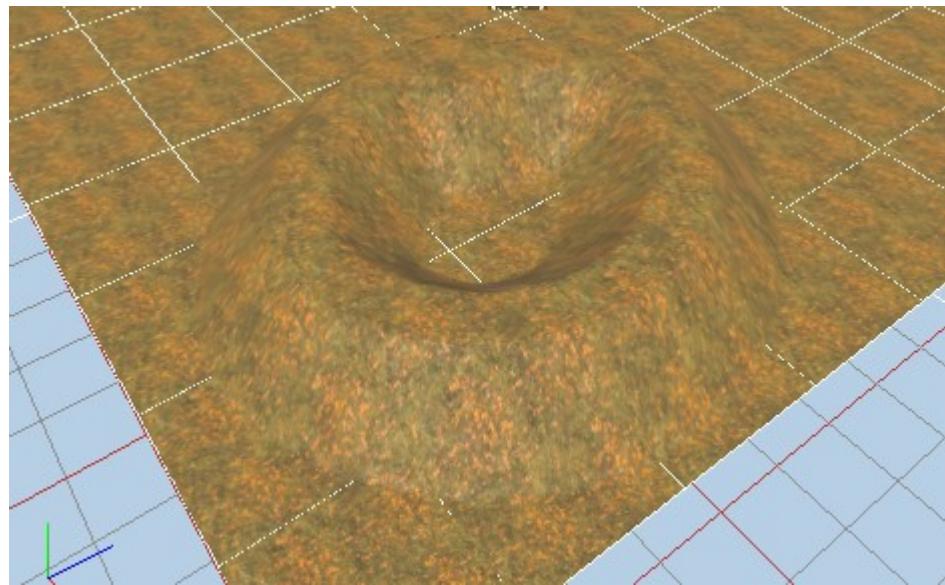
Left click on the **(Geometry)** element in the **Scene View**. This pops up a **Brush** dialog box. It's called a "brush" because you will be "painting" your terrain height changes onto the map. You can move the **Brush** dialog box to the side if you don't want it to obscure the main panel.



By default, your paintbrush shows up as a red dot on the terrain. Below I've increased the size and value of the brush to make it stand out more.



When the brush is active, you can zoom the camera in and out with the scroll wheel and rotate the camera with the left mouse button as usual. In addition, you can now adjust the terrain height with the right mouse button. With the mouse over the terrain in the main panel, hold down the right mouse button while moving the mouse. The terrain will increase in height by the amount of the brush **Value** if the value is positive, and it will decrease in height if the value is negative.



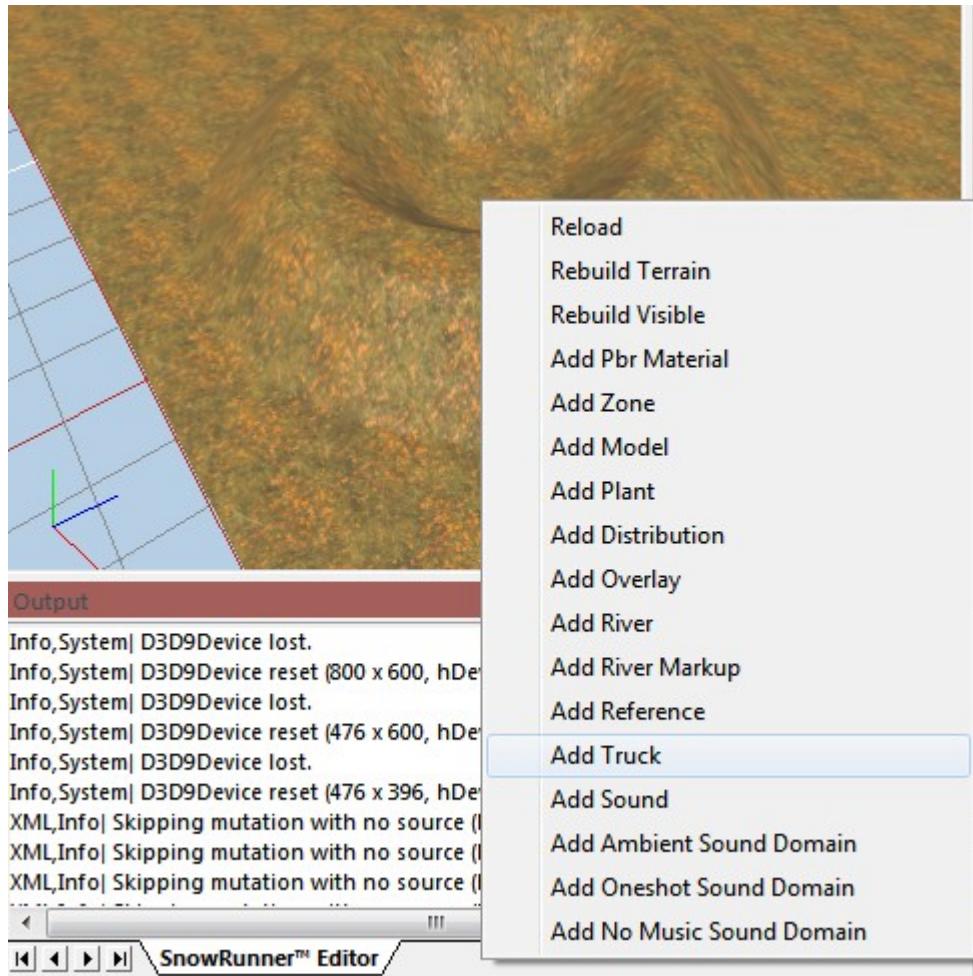
To make larger changes to the terrain, you can adjust the size and value of the brush, or you can make multiple passes with right click and drag.

When you are done editing the terrain height, left click in the main panel to exit painting mode. Note that left click and drag moves the camera while remaining in painting mode, while left click with no mouse movement leaves painting mode.

Place a Truck

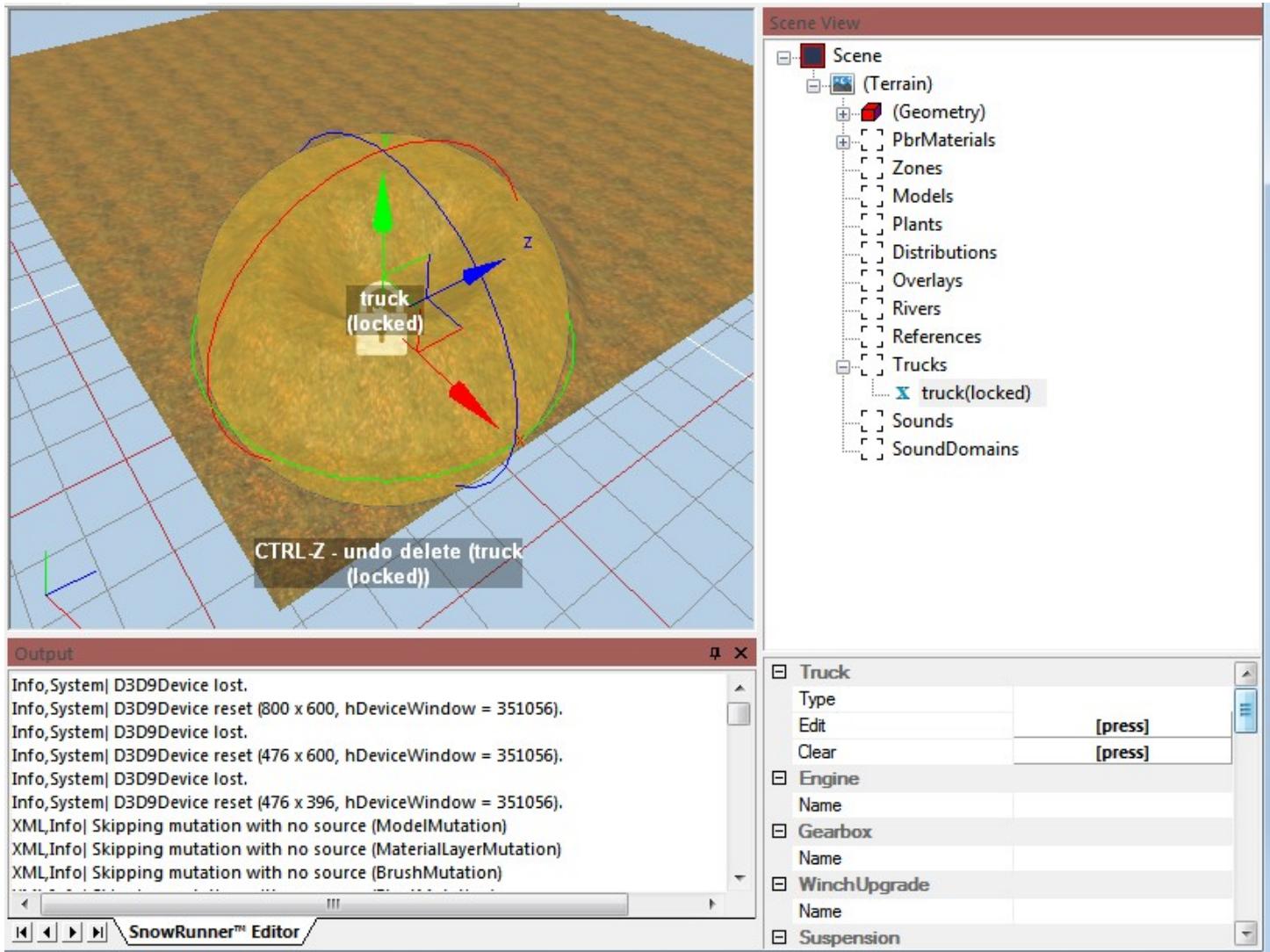
Now that you have some interesting terrain, you'll need a truck to drive around in it.

If necessary, left click to exit painting mode. Then point the mouse over the terrain in the main panel, right click, and select Add Truck from the context menu.

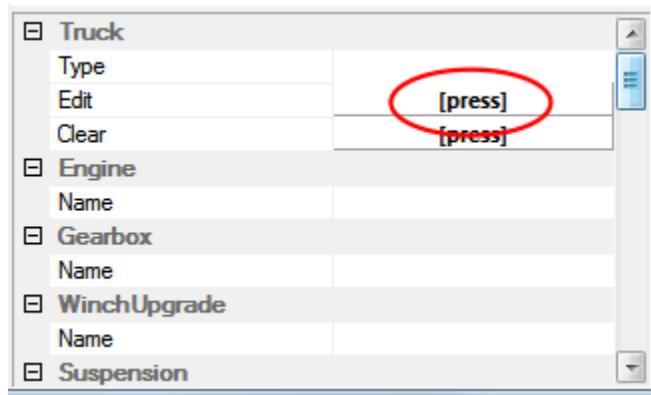


Bug: Unlike in most other programs, the context menu will not appear until you release the right mouse button.

A new truck appears in both the main panel and in the **Scene View** panel. Below the **Scene View** panel is an unlabeled panel that shows various parameters for the truck. I call this panel the property panel.



The truck that you've added doesn't yet have a type. To tell the editor which kind of truck it is, click in the property panel next to [Truck→Edit](#) where it says [\[press\]](#).



Bug: The first time you click here, the editor may lock up for 5 or more seconds as it creates icons for all possible trucks. It will be faster for subsequent uses.

The editor pops up a dialog listing all of the available kinds of trucks, as well as other things (such as trailers) that the game puts in the truck category.

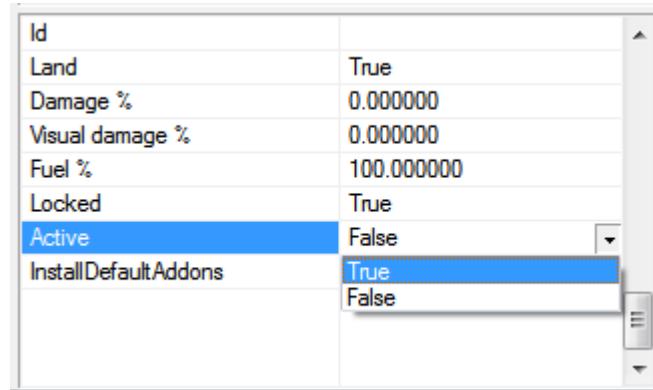


For testing purposes, I like the [ank_mk38](#) because it is a very capable truck in its base form, and because I don't have to scroll to find it. Click to select the truck that you like and click [OK](#).

If you have experience with the MudRunner Editor, you may remember that changed feature properties did not commit until you changed the selection (e.g. by clicking elsewhere). However, the SnowRunner Editor mostly fixes this bug so that property changes take effect immediately. The few exceptions where the commit is delayed are described where appropriate.



There is one more step to make this truck drivable. If necessary, re-select the truck by clicking it in the **Scene View** panel. Then scroll to the bottom of the property panel. Where it says **Active False**, click on **False**, and change the dropdown value to **True**.

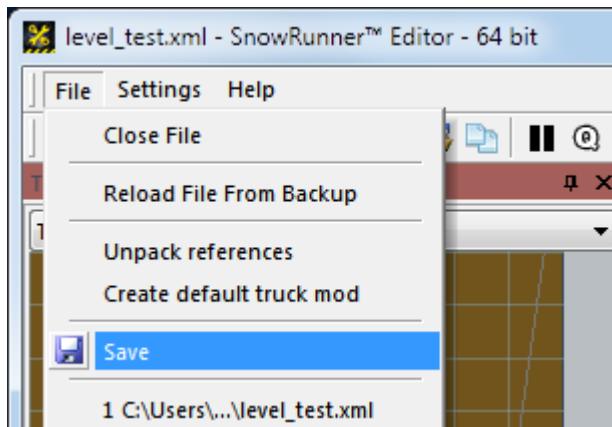


The truck is now active.



Save and Pack

To save your work so far, select [File→Save](#) from the top menu bar or press the keyboard shortcut: [Ctrl-S](#).



Now pack your map for testing. Click the stack of books in the toolbar. The toolbar is just under the top menu bar, and the stack of books is the fourth icon from the right end.



A dialog box warns that packing the level takes time. Click [Yes](#) to continue and wait for it to complete. For a map this simple, it will only take a few seconds.

Packing your map saves it automatically, so the previous save step wasn't strictly necessary.

Make a Backup Copy of your Campaign

In the course of working with the Editor, you will create a lot of new games for testing maps. SnowRunner gives you four save slots, so hopefully you have room to create these new games without disturbing the save files for your campaign, but there is always the possibility that you make a mistake.

Bug: Even worse, on a couple of occasions I've left SnowRunner idling on the main menu in the background only to find out that it had spontaneously created a new game in a random save slot!

To avoid losing your campaign progress, I recommend making a copy of your data.

If you bought SnowRunner from the EPIC Store, your save files are in `%USERPROFILE%\Documents\My Games\SnowRunner\base\storage`.

If you bought SnowRunner from Steam, your save files are in a path something like `C:\Program Files (x86)\Steam\userdata`. That directory has a randomly numbered subdirectory. This subdirectory contains directories for each of your Steam games which are also unhelpfully labeled with random strings of digits. If you've played SnowRunner recently, then its folder should be one of the most recently modified. Within that directory, your save data is stored in the `remote` directory.

Make a backup copy of all files in this directory. If you ever need to restore some files from backup, you can either restore them all, overwriting all four save slots, or you can pick and choose. But you definitely want to archive all of the files now so that you don't discover later that you missed an important one.

Test Your Map

Time to take your map for a spin. Exit the SnowRunner Editor and start the SnowRunner game.

Tip: You have to wait through the epilepsy warning every time, but you can skip the rest of the ads by pressing the `Escape` key as soon as the epilepsy warning starts to fade out.

From the game's main menu, select `New Game` and then `Custom Scenarios`.



The game will list all available mods. Most likely this is only the one you created, e.g. [mod_level_test](#). Select your map and a save slot for your custom game. The save slot must be separate from your save for the main game because progress for each mod is saved separately.

Bug: The game will not reliably replace an existing save with a new game. It asks for confirmation as if it is replacing the existing save, but then it loads the old game instead of the new game.

As far as I can tell, this bug always occurs when a new game is the first game loaded after SnowRunner is started. Exit the (faulty) game and try to create a new game again, it works.

I often leave SnowRunner running in the background as I repeatedly make edits to a map and test it. That means I don't have to wait for the game to start up, **and** it will work properly when I create a new game.

The game now starts in the truck that you chose and in the terrain that you drew.



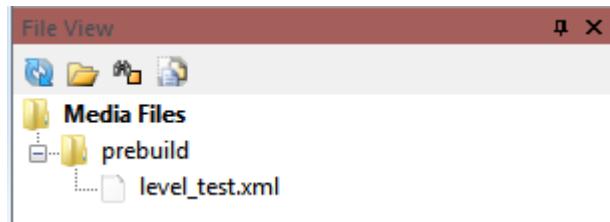
The game controls are all as usual, with one exception. There is now a permanent menu of **Tools** in the upper right of the screen. The **Tools** menu provides useful actions for testing such as spawning new trucks, allowing truck modification as if you are in a garage, etc. See the Dev Tools section on page 150 for more information about this menu.

Bug: The default grass and leaves texture on the terrain looks unnaturally large compared to the size of your truck. That can be fixed by manually adjusting the scale of the PbrMaterials. Recommended scale values are listed in section TBD.

Re-editing a Map

If you quit and restart the SnowRunner Editor, it will not automatically return to the map you were previously editing.

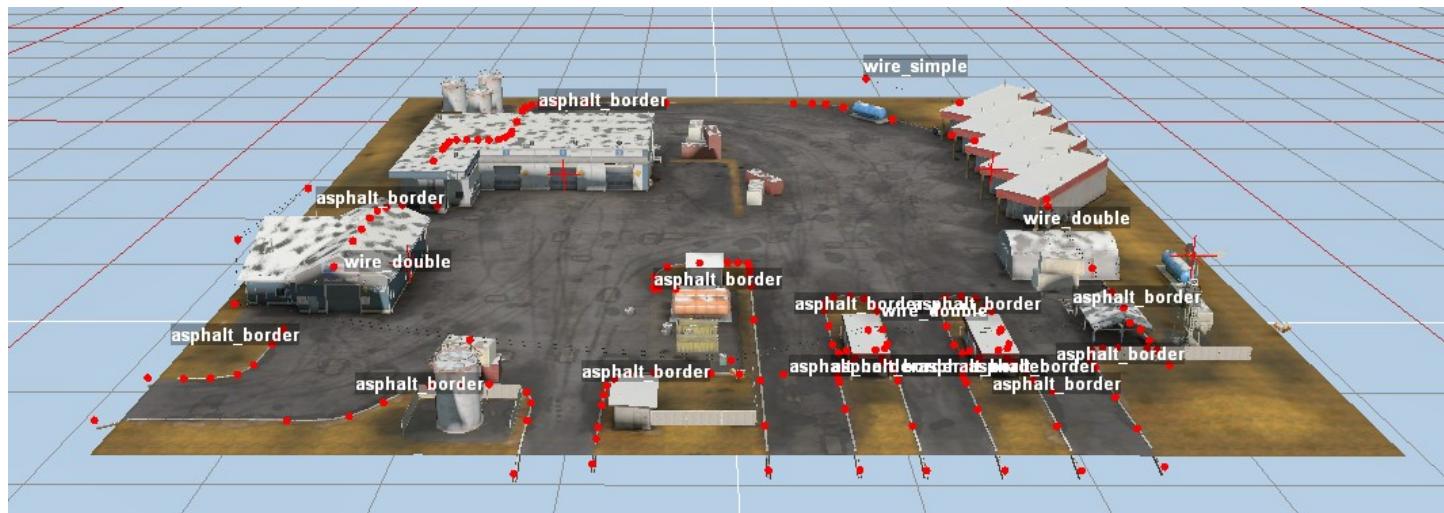
The maps that you have created are listed in the `File View` panel under `Media Files→prebuild`. To edit a map, double click it.



Example Maps

Saber has not published any of their maps in an editable format. However, their maps use a number of shared map sections that Saber has kindly made available for modders to use. `File→Unpack references` unpacks these map sections from the installation directory and puts them in the `Media/prebuild` directory where they can be read into the SnowRunner Editor.

Bug: `Unpack references` takes a very long time (over a minute on my system) with no indication of progress. Be patient, or watch the files as they appear in the `Media/prebuild` directory.



How to Use This Book

Sources of Information

This book is designed so that you can read it straight through from beginning to end. Each section builds on the information from previous sections to form a complete picture without too many instances of “I’ll tell you later”. However, you can also just start experimenting with the editor while reading only the sections of this book that strike your interest. To accommodate this, I’ve done my best to label all sections so that you can easily find the section you need in the table of contents.

I should point out that this is not your only source of information regarding SnowRunner map making. There are various other guides on the web in either text or video form, and Saber have themselves published a reasonably complete guide which you can find installed at a path similar to this:

```
C:\Program Files (x86)\Steam\SteamApps\common\SnowRunner\Sources\BinEditor\Guides\  
SnowRunner_Editor_Guide.pdf
```

Advantages of the Big Book of SnowRunner Map Making:

- Faster “quick start” to editing and testing.
- Includes topics not included in the Saber guide.
- Covers all topics in exhaustive detail.

Advantages of the Saber guide:

- More likely to stay up to date as the game and editor are updated.
- Not as exhausting to read.

By the way, if you think this book is useful, please give it a “thumbs up” wherever you found it. Those thumbs up will help more people find and enjoy this book. More educated modders means better mods!

Is This Book Up To Date?

There are a lot of websites that will “helpfully” copy this book for their readers, but won’t keep it up to date. The official release location is on GitHub: <https://fred-rum.github.io/bbsmm/BBSMM.pdf>

The title page of this book has the date on which I last edited it. However, Saber doesn’t notify me when they change something, so it’s quite possible for me to edit the book after a change and yet not include it.

I’ll make announcements in the Steam forums and Focus forums when I make major updates. If you notice anything missing or wrong, let me know in the comments to these announcements.

Editor and Game Bugs

No mod editor will ever be designed to be as consumer friendly as the game itself for a number of reasons:

- Only a fraction of game buyers will even try mod editing.
- People who edit mods tend to be more tech savvy and better able to work around any bugs that exist.
- The universe of possibilities is much larger when editing mods than when playing a game. That makes it harder to test every corner of the design and iron out the bugs.

In most cases, a mod editor is tested only to the extent that it was used to make the game levels themselves. If a feature wasn't used in the game, then it probably wasn't tested in the editor.

If it sounds like I'm making excuses for Saber, I totally am. I have complete respect for Pavel's original Editor and for all of the quirks that Saber has smoothed out of it. But Saber has also added a pile of new features which come with their own bugs.

For any insufficiently tested feature, there can be a bug in the Editor (so the Editor can't edit that feature properly) or a bug in the game (so that the game doesn't work correctly with that feature). I've chosen to highlight either type of bug as follows:

Bug: This is an example bug warning. I'll explain when the bug occurs and how to recognize it, how to avoid it (if possible), and how to fix any problems it causes (if possible).

I generally put each bug warning immediately after the related text that describes how to use a feature. So if you see a bug warning, read it!

And if Saber wants to use these bug reports while developing the next version of the Editor, I wouldn't mind that, either. :)

Design Tips

I'll admit right now that I am not the best map designer. I'm an engineer at heart, and my artistic eye is limited. Therefore, I'm hesitant to give a lot of design tips in this book. Where I do give a tip, it is often highlighted as follows:

Tip: This is an example design tip. You don't have to do it this way, but this is probably a good way to do it.

Some of these tips I've directly borrowed and expanded from Saber's guide. Other tips come from my experience with making maps and/or wrestling with the Editor for long periods of time. I try to stay away from

totally obvious tips. You already know what a natural world looks like, so I'll leave it up to you to decide whether rocks should go in trees.

If You Made Maps in MudRunner

If you made maps in MudRunner, then the editor should be mostly familiar to you. I recommend reading through the Quick Start and Directories, Files, and Archives sections since those highlight some important differences from MudRunner.

In particular if you already read my Big Book of MudRunner Map Making, you'll find a lot of familiar material in this book. On the other hand, a lot of things have changed from MudRunner, and a lot of things are new. If you have the patience for it, I recommend at least a skim through the entire book to avoid frustrations later.

The MudRunner Editor included the ability to import maps from SpinTires. However, the SnowRunner Editor doesn't include any ability to import maps from SpinTires or MudRunner. You may be able to import selected features of previous maps by manually copying bitmaps and XML sections, although you'll need to hand edit the names of any assets you used such as models, overlays, and distributions. I haven't tried it, so I can't estimate your odds of success.

Directories, Files, and Archives

It is useful to understand the directory structure used by the SnowRunner Editor since you will need to occasionally interact with it.

The primary directories of interest are in `%HOMEPATH%\Documents\My Games\SnowRunner\Media`

Prebuild Directory

The `prebuild` directory contains the input files required to build your maps. For each map, there is a corresponding XML file and a directory. The directory starts with a number of default files that will be updated as you edit your map. More files may also be added, depending on which editor features you use.

In the below example listing, I have organized the files and directories by type:

```
prebuild/
    level_test.xml
    level_test/
        _sounds.xml
        _zones.xml
        _height.dds
        _height_source.png
        _base_tints.tga
        _merge_mud_wetness.tga
        _quickmud.tga
        _snow.tga
        _tint.tga
        _water.tga
        _water_flow.tga
        _water_mud.tga
        mtrl_base.blends.tga
        mtrl_layout.tga
        mud
        subgroups/
        ui/
            textures/
```

You generally won't ever need to manually edit these files outside the SnowRunner Editor. However, I briefly explain their format here in case you want to do something that is easier to do outside the Editor than inside it.

The XML files are a standard ASCII/UTF format which can be easily read and edited by humans. These files start out as mostly empty stubs, with XML tags ready for new content to be added. The `<map_name>.xml` file contains most of the non-bitmapped data that describes your level, but some information is split off into the `_sounds.xml` and `_zones.xml` files.

The `_height.dds` file is in an old Microsoft DirectDraw bitmap format. Not many graphics editors can read and edit this format anymore. Interestingly, `_height_source.png` is **also** in DDS format, despite its suffix. In general, when SnowRunner reads a file containing a bitmap, it ignores the file suffix and determines the format from the file contents. This can be handy sometimes, but it can also be deceptive.

The `*.tga` files are also bitmaps in an old format created by Truevision. Fortunately in this case, many modern graphics editors (e.g. the GIMP image editor) can read and write this format.

As far as I can tell, the `mud` file is a collection of images in the DDS format. Because SnowRunner's mud model is highly detailed, the `mud` file holds data only for terrain blocks where mud is present. So it starts out very small, but it can expand to very large if your map contains a lot of mud.

The `subgroups`, `ui`, and `textures` directories start out empty.

If you delete the `prebuild` directory or its contents, then you will lose all of your work. Unless you follow the recommendation in the next section...

Create an Archive

Because your mods are saved in your user area and not the game area, your mods are safe even if you uninstall SnowRunner. However, there is always the possibility of losing or corrupting your files either through user error or an editor bug. For this reason, keeping archives of your work is important.

The easiest archival method is to occasionally copy the necessary files and directories from the `prebuild` directory to another directory. Even better, copy it to a different drive in case you have a drive failure.

Bug: Some files necessary to rebuilding your map are stored in the `levels` directory. See the next section for details.

Levels Directory

The `levels` directory contains the files required by the game executable. If you think of the files in the `prebuild` directory as the “source” files, the files in the `levels` directory are the “compiled” files.

The `levels` directory contains a directory corresponding to each of your maps. If you have packed a map, then there is also a ZIP archive that contains your packed mod, ready for uploading to `mod.io`.

```
levels/
  level_test/
    data.stg
    ...
  level_test.zip
    level_test.pak
    level_test_playstation_4.pak
    level_test_xbox_one.pak
```

The `data.stg` file contains the majority of the map data, stored in an opaque binary format. The remaining files in the `<map_name>` directory are more or less ready to be manually edited, but there isn't any reason to do so.

The `*.pak` files are themselves each an archive of files that can be opened with a supported archive manager such as 7-Zip. These files are essentially just a repackaging of the same files as are in the `<map_name>` directory.

If you delete the `levels` directory or its contents, then the SnowRunner Editor will offer to recreate the data as you re-open each map for editing. The ZIP archive will be re-created when you next pack your map.

Bug: Some files necessary to rebuilding your map are stored in the `levels` directory instead of the `prebuild` directory. So when you archive the `prebuild` directory, also make a copy of these files from the `levels` directory if they are present: `content_settings.json`, `level_desc.json`, `objective_settings.json`, and `zone_settings.json` (i.e., all `*.json` files).

Mods Directory

The `Mods` directory contains the same `*.pak` files as above for each map, as if they had been downloaded as a mod. These are the files that are read by the SnowRunner game.

If you delete the `Mods` directory or its contents, the `*.pak` files will be re-created when you next pack your map.

AppData Directory

SnowRunner keeps track of past activity in a separate area: `%appdata%\SnowRunner 2 Editor`

E.g.

```
backup/
  level_test/
    ...
CustomColor.xml
LastStates.xml
Log.txt
ViewTerrain.xml
```

The `backup` directory keeps for each map a number of backups of the XML files only, not the bitmap files. These can be restored through the editor as described in the Menu Bar section on page 29. Or the editor may offer to restore a backup automatically if it detects a problem.

The `CustomColor.xml` file records any custom colors that were set in the color picker, e.g. in the brush dialog for `(Geometry)→(Colorization)`.

The `LastStates.xml` file is used to detect when the editor crashes (because the file isn't updated correctly before closing).

The `Log.txt` file keeps the text from the Output panel for all sessions. (So this file eventually gets quite large.)

The `ViewTerrain.xml` file keeps for each map the most recent camera position within the editor. If the editor camera gets hopelessly lost, quit the editor, delete this file, and restart.

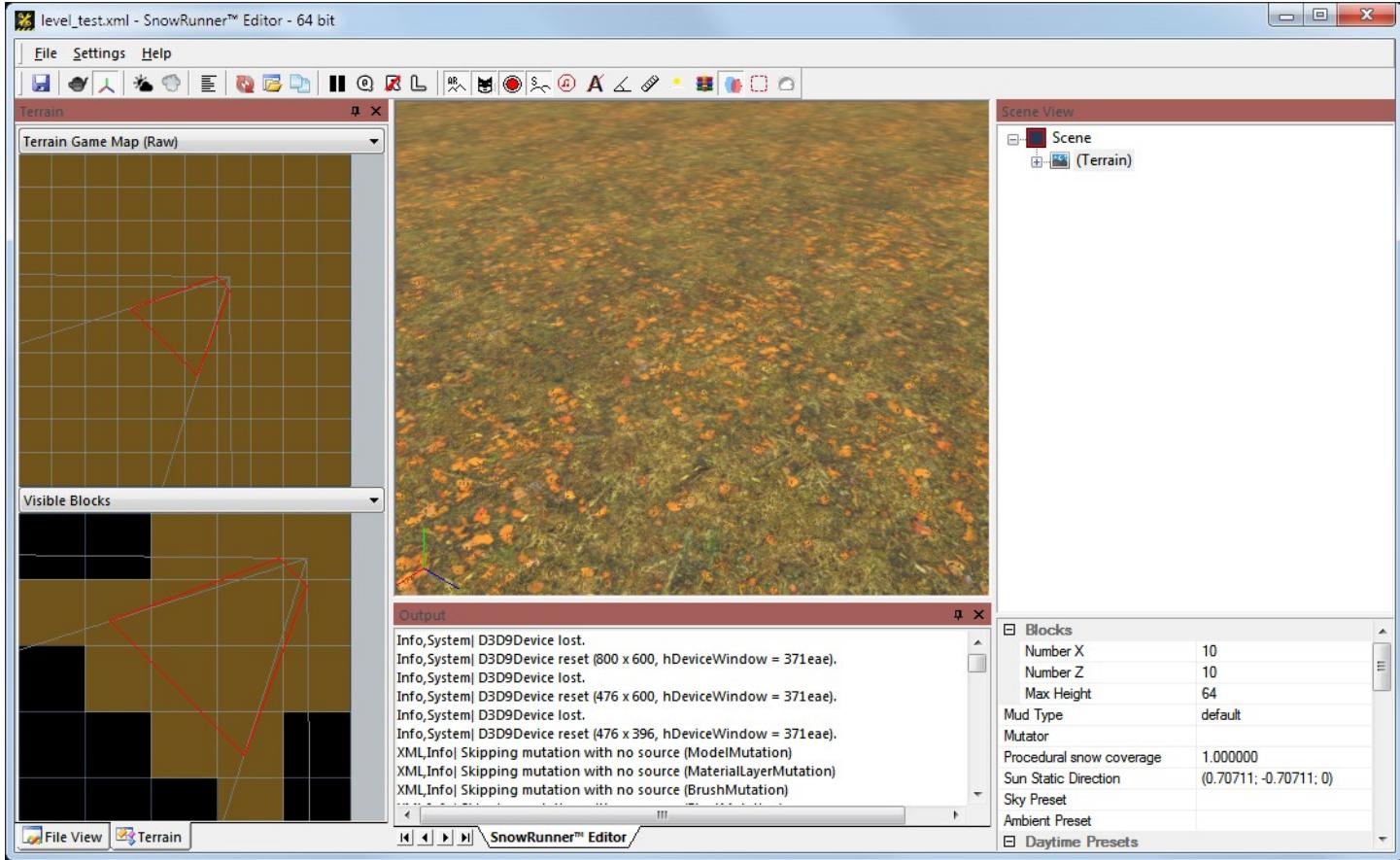
Config Directory

SnowRunner keeps track of the editor configuration here: `%HOMEPATH%\Documents\My Games\SnowRunner\base\config\editor.cfg`

Currently, however, the only saved information is whether `Show Snow By Up Vector` is enabled.

The Map Editor Window

After creating a new map, the map editor opens. It contains a number of sections that I'll go over briefly here.



Menu Bar

The menu bar at the top of the window has **File**, **Settings**, and **Help** menus.

File Menu

Possible actions in the **File** menu are as follows.

Close **File** closes the map and returns to the **File View** panel. If there are unsaved changes, the Editor will ask if you want to save first.

Bug: Edits in the property panel are not considered to be unsaved changes until the edits are committed by changing the selection, e.g. by left-clicking elsewhere.

Reload File From Backup opens a dialog that allows you to restore XML files from a selected backup. Note that bitmap files are not backed up and cannot be restored from backup. The location of the backups is described in the AppData Directory on page 27. I do not know how often backups are recorded.

Saber used a large number of map reference sections that they have kindly made available for modders to use. The Example Maps section on page 21 describes how to use **Unpack references** to get these references.

Create default truck mod is used for creating truck mods. Truck mods are not covered in this book.

Save (shortcut: **Ctrl-S**) saves the map's the named properties of all features from memory to disk. This differs from bitmaps painted with a brush, which are updated on disk as soon as a change is committed.

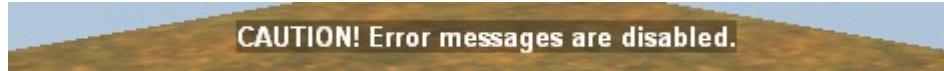
Exit exits the Editor.

The **File** menu also includes a list of recent edited maps. Selecting closes the current map and opens the selected map.

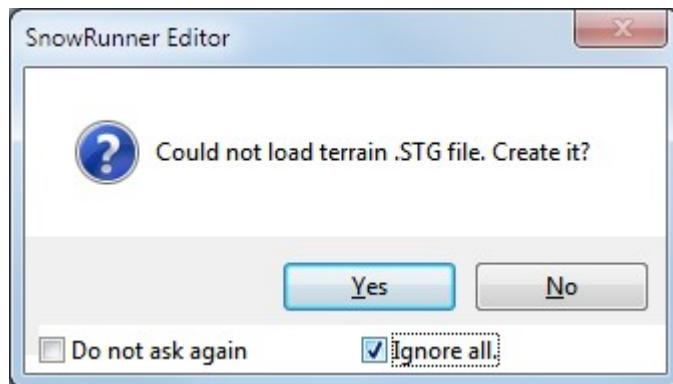
Settings Menu

Possible actions in the **Settings** menu are as follows.

Ignore Warnings disables all warning dialog boxes. The editor prominently warns you when warnings are being ignored.



The same can be accomplished by selecting **Ignore All** in a dialog warning box.



The **Ignore Warnings** setting applies across all maps, but it **does not** persist across Editor sessions.

Show Snow By Up Vector shows the snow cover effect for models and plants in snowy areas. This setting applies across all maps and **does** persist across Editor sessions.

Help Menu

The only option in the **Help** menu is **Guides**. This opens a Windows Explorer window for the folder containing the official guide PDFs that were installed with the game.

Toolbar

Under the menu bar is a toolbar with a row of buttons for various features. You can hover over each button to get a description of what it does.



Many of the buttons are toggles. Click once to enable, click again to disable. The Editor always starts with the same default mix of enabled and disabled buttons regardless of what was selected in the previous session.

Save (shortcut: **Ctrl-S**) saves the map's the named properties of all features from memory to disk. This differs from bitmaps painted with a brush, which are updated on disk as soon as a change is committed.

Wireframe (shortcut: **Ctrl-W**) is a toggle button. When enabled, it redraws everything using only colored outlines for all triangles, instead of filled shapes.

Grid (shortcut: **Ctrl-G**) is a toggle button. When enabled, it draws a reference grid below the terrain at height 0. The size of each square depends on the zoom level, so it can't be used as a measurement tool, but it is sometimes helpful to keep yourself oriented.

Night is a toggle button. When enabled, it changes the lighting to night mode. This is useful for testing how everything looks in the dark, especially lights.

Fog has is a toggle button with no known effect.

Statistics (shortcut: **Ctrl-T**) is a toggle button. When enabled, it puts a block of text in the upper left detailing the camera position and orientation and statistics regarding what is being drawn.

Reload Resources reloads most data used by the map (as opposed to data **about** the map). E.g. this includes shapes, texture bitmaps, etc.

Open File (shortcut: **Alt-Shift-O**) pops up a dialog box listing all files in your Media directory. When you select a file from this list, the Editor attempts to open the file in some manner:

- If you select a map or mesh XML file, it opens in the SnowRunner Editor. These can also be opened via the File View (page 36)
- If you select another XML file, it opens in whatever editor Windows lists as your default (usually some sort of text editor).

- If you select a bitmap/texture file, the Editor pops up an error dialog requesting that you specify a texture editor path in the Editor settings. However, although the MudRunner Editor had such a setting, the Snow Runner editor does not, so the error message now just looks silly.
- Finally, if you open some other file type, the Editor pops up an error dialog to say that it isn't supported.

Duplicate selected (shortcut: **Ctrl-D**) makes a copy of the currently selected feature.

Pause is a toggle button. When enabled, animation of models is paused. (E.g. `wind_mill_01` is an animated model.)

Quick mode is a toggle button. When enabled, rebuilding the terrain causes the Editor to rebuild large features as usual, but it hides grass, shadows, and other minor features rather than rebuilding them. This can speed up the rebuild quite a bit, especially on large maps.

No references is a toggle button. When enabled, rebuilding the terrain causes the Editor to hide references instead of rebuilding them. References are described in TBD.

Local transform is a toggle button. It is described on page 58.

Enable autorebuild terrain is a toggle button. When enabled, the terrain is rebuilt whenever you commit a change to a (**Geometry**) feature. Major changes are updated in real time as you edit, but shadows, 3-D grass, and other minor features are updated only after the edits are committed (and only if this button is enabled).

Show all sound domains is TBD.

Hide Strings is a toggle button. When enabled, it hides the labels that normally display near each object.

Show HeightMap Angles is a toggle button. When enabled, it recolors the terrain to show the angle of the terrain at each point. Further information is in the Height Information section on page 73.

Use Ruler is a toggle button. When enabled, a line segment or complex curve can be overlaid on the terrain, and the Editor reports its length. More detail is in TBD.

Terrain brightness pops up a dialog box where the brightness of the terrain can be adjusted. Other map features are not effected. This is particularly useful on snowy maps with a lot of white terrain.

Pack terrain converts the map to a format that can be played in the game and uploaded to mod.io. More detail is in TBD.

Collision Notification is a toggle button. When enabled, dynamic models are highlighted if they collide with the terrain, other models, or individual plants. More detail is in TBD.

`Zone settings` opens the Zone Settings Editor. More detail is in TBD. **Caution:** while the Zone Settings Editor is open, the main SnowRunner Editor window is completely unresponsive. Close the Zone Settings Editor to return to the main SnowRunner Editor.

`Region settings` opens the region settings editor. More detail is in TBD.

Main Panel

Most of your time will be spent in the main panel. This panel is unlabeled, but you can find it centered between the `Terrain` panel on the left and the `Scene View` panel on the right. It shows a “camera eye” view of your map.

Navigate in the Main Panel

The main panel shows the 3-D map from the perspective of a virtual camera. To view different parts of the map, the camera can be moved in all three dimensions, it can be panned left and right, and it can be tilted up and down. To prevent disorientation, the camera is never tilted left or right.

A small icon in the lower left of the main view shows the coordinate axes as they appear from the current camera angle. The red line shows the X axis and always points east. The green line shows the Y axis and points up. The blue line shows the Z axis and points north. The SnowRunner coordinate system is described in more detail in the TBDError: Reference source not found section on page Error: Reference source not found.



You can move the camera around as follows:

- left click and drag – Move the camera around whatever terrain you clicked on and rotate the camera to continue pointing at that spot. This is my favorite method for rotating the camera. If the initial click is not on visible terrain, the anchor is attached to the 0-elevation reference plane instead.
- mouse wheel up or down – Push the camera closer to the terrain under the mouse or pull it further away. (Caveat: the most recent click or drag must have been in the main panel.)
- ctrl + left click and drag – Keep the camera pointed the same way while moving it relative to whatever terrain you clicked on. This effectively drags the terrain around under the mouse, and it’s my favorite method for moving the camera across the map.
- shift + left click and drag – Keep the camera fixed in space and rotate it to point in new directions.
- alt + left click and drag – If a feature is selected, move the camera around that feature and rotate the camera to keep that feature in the same spot in the view (or off view). If no feature is selected, the alt modifier is ignored.

It is very easy to get completely lost in the main panel so that no terrain is visible. If the camera is too far away from it, the display engine won't draw it. If that happens, double click in the **Terrain** panel to recenter your camera over the terrain.

If the camera is below the terrain, the display engine also won't draw it. In that case, double clicking in the **Terrain** panel will move the camera close to the terrain, but still pointing up at it, so the terrain is not drawn. In that case, left click in the main panel and drag downwards to tilt the camera downward while moving the camera itself upward until the terrain comes into view.

Extremely rarely you may end up with the camera pointed straight up or down, and left click and drag won't rotate it back to a proper angle. The editor tries to prevent this by keeping the camera just off of vertical, but its algorithm can occasionally fail. Shift left click and drag may work when other methods fail. Otherwise, you'll need to quit the editor and edit the file that stores the camera position:

```
%appdata%/SpinTiresEditor/ViewTerrain.xml
```

Find your level in the file and delete its entry. The next time you open the level, the camera will reset to its default position and angle. (This may be underground, but that's more easily fixable.)

Select Targets in the Main Panel

You can select one or more targets in the scene as follows:

- left click – Select the terrain block, model, plant, or spline point under the mouse. Other types of objects cannot be selected in the main panel. This notably includes trucks.

Bug: A model, plant, or spline point cannot be clicked on in the main panel if it is off the edge of the map.

- ctrl + left click – Add the terrain block, model, or plant to the current selection or remove the terrain block from the selection. It is not possible to remove a model or plant from the selection in this manner. It is also not possible to select different types of targets at once. E.g. a model and a plant cannot be in a selection together. Ctrl-clicking a different type of target resets the selection as if it were clicked without the control key.
- double left click – Select the terrain block, model, plant, or spline point under the mouse and move the camera (without rotation) to center the selected item in the scene.
- right click and drag – Select all models within the dragged rectangle. (This only works for models.)

Other Actions in the Main Panel

A right click without a drag brings up a context menu. The context menu is described on page 47.

A middle click (e.g. depressing the mouse wheel) displays a copy of the coordinate axes at the mouse position for as long as the middle button is held down. It performs no other action.

Output Panel

The **Output** panel (below the main panel) shows logging information and error messages from the editor and the embedded display engine. You will refer to it occasionally when things go wrong.

Note that the **Output** panel will often contain lines saying “D3D9Device lost” and “D3D9 Device reset”. These messages look ominous, but they seem to be part of normal operation, and you can ignore them.

At the bottom of the **Output** panel are four arrows pointing left and right. I do not know what these do.

The angled tab at the bottom of the **Output** panel indicates that it is showing the main log from the SnowRunner Editor. On occasion, another tool within the Editor may create a separate tab in the **Output** panel in order to display its log. You can then click on the angled tabs to switch between logs.

Terrain Panel

The Terrain panel on the left includes two subpanels looking into the scene from a top-down view.

By default, the upper panel shows a view of the entire map. A truncated triangle shows the field of view of the main panel’s camera. If the map is not square, it is stretched into a square for display in the terrain panel.

The terrain panel only shows the colors that are painted on the terrain itself. So, for example, it will show a road overlay, but not a house model. If the house model has shadows, the terrain panel will show one type of shadow but not the other.

A dropdown menu above the upper panel changes it to show many other map textures used by the display engine. I don’t find these to be very useful. Once you’re done looking around, flip it back to the default **Terrain Game Map (Raw)** at the end of the dropdown list.

The lower panel is similar to the top one, but by default it zooms in on the terrain blocks that are visible to the camera. (I’ll describe terrain blocks later in the guide.) A truncated triangle again shows the field of view of the main panel’s camera. Only those terrain blocks within this view (or very near it) are drawn in the lower panel.

Another dropdown menu allows the lower panel to be switched to show only the **Selected Blocks**. I find the **Visible Blocks** default to be the more useful view for the lower panel.

Changes made from the dropdown menus in the Terrain panel do not persist across Editor sessions.

Below the two map panels is a block of text that gives various information about the selected terrain blocks. The information isn't particularly useful, though, so I never pay any attention to it.

Navigate in the Terrain Panel

Double click a terrain block in either part of the terrain panel to fly the camera in the main panel to that block. (This also selects the block.)

It is tempting to rotate the scroll wheel up and down within the terrain panel to zoom in and out, and it sort of works. But because the mouse is positioned outside the main panel, the target point for the zoom is off the edge of the view, so zooming in isn't very effective. Move the mouse back to the main panel before zooming.

Select Targets in the Terrain Panel

Left click in either part of the terrain panel to select a terrain block. Hold the **Control** key while you left click to add or remove a terrain block from the current selection.

File View Tab

At the bottom of the **Terrain** panel are two tabs which you can use to switch between the **Terrain** panel and the **File View** panel.

The **File View** panel lists selected files and directories in the **Media** directory.

Double-click any map's XML file in the **prebuild** folder to open that map. Only one map can be open at a time, so if you have a map open already, the editor will prompt you to save it before opening a new one.

Double-click a folder to expand or contract its level of hierarchy.

Right-click the **prebuild** folder or any sub-folder for a context menu that allows you to create a new directory (**New Folder...**) or a new map (**New Terrain...**) in that folder.

Remember that you can open a recently edited map by selecting it from the File menu, but you can open any map at all by double clicking its XML file under the prebuild directory for its mod.

The **File View** tab has its own toolbar. You can hover over each button to get a description of what it does.



The **Update** button refreshes the file list.

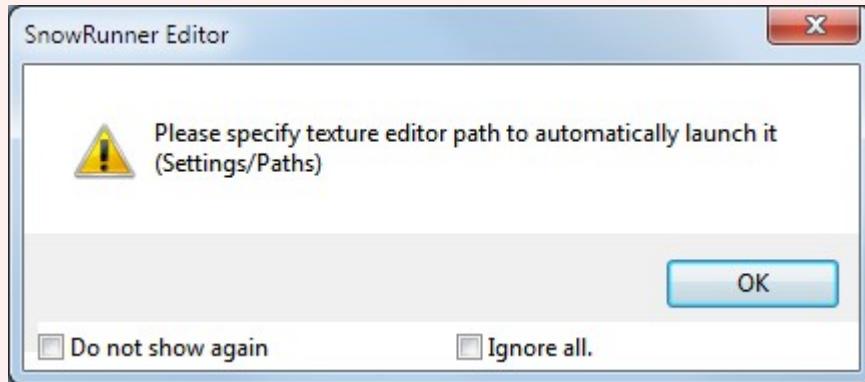
The **Show in Explorer** button opens the selected folder in Windows Explorer.

The **Find References** button finds all references within the **Media** directory to the selected XML file. For example, it can find all maps that use a reference map, or it can find all references to a custom asset. The **Find References** button remains depressed while the search is ongoing, and it releases when the search is complete. The search results are displayed in the **Output** panel in a new tab called **Find References Results**. Be aware that the **X** button in the corner of this tab closes the entire **Output** panel, not just the tab.

The **Show All Files** button is a toggle. When enabled, it shows all files and folders in the Media directory. In the default disabled state, it shows only the top-level XML file for each map, and folders are shown only if they contain at least one top-level XML file.

When all files are shown, double-click an XML file to open it in your default XML editor (e.g. Notepad++).

Bug: When all files are shown, if you double-click a bitmap file, the Editor complains that no texture editor has been configured. This is a holdover from the MudRunner Editor; there is no way to specify the path to a texture editor in the SnowRunner Editor.



Controls for Hierarchical Lists

The **File View** tab, the **Scene View** panel, and the property panel all display hierarchical lists that share a number of controls in common.

A few common controls are performed by the mouse:

- Left click item selects that item.
- Left click on a **+** to the left of a folder (in the **File View** tab) or hierarchical container (in the **Scene View** panel or property panel) expands that container, showing its contents.
- Left click on a **-** to the left of a folder or hierarchical container contracts that container, hiding its contents.
- Double click on a folder (in the **File View** tab) or hierarchical container (in the property panel) expands or contracts that container. Double click has no effect in the **Scene View** panel.

- Right click performs special actions specific to the panel in question.

Certain actions can also be performed using the keyboard:

- The up and down arrows move the selection up and down the list.
- The right arrow does different things in different contexts:
 - On an unexpanded folder or container, it expands the container.
 - On an expanded folder or container, it moves to the first item in the container.
 - On a non-container item in the property panel, it moves the selection down to the next item.
 - On a non-container item in the **File View** tab or **Scene View** panel, it does nothing.
- The left arrow does different things in different contexts:
 - On an expanded folder or container, it hides the items in the container.
 - On an unexpanded folder or container, it moves to the container of the current item (up one level in the hierarchy).
 - On a non-container item in the property panel, it moves the selection up to the previous item.
 - On a non-container item in the **File View** tab or **Scene View** panel, it does nothing.
- The **Home** key moves the selection to the top of the list.
- The **End** key moves the selection to the bottom of the list.
- The **Page Down** key moves the selection down by approximately one page.
- The **Page Up** key moves the selection up by approximately one page.

In all cases, the panel automatically scrolls to keep the selected item in view.

Scene View Panel

The **Scene View** panel on the right contains a categorized list of all the features on the map.

Select Targets in the Scene View Panel

Left click on a feature or category selects it.

Ctrl + left click on a feature in the same category as previous selections adds that feature to the selection. Features in different categories cannot be selected at the same time.

Ctrl + left click on a selected feature removes it from the selection.

Shift + left click on a feature in the same category as a previous selection adds all features from the initial selection to the clicked feature to the selection.

Bug: Shift click does not correctly highlight the selected items, although the proper items are secretly selected. If you're not sure, shift click the same item again, and all items should highlight correctly.

Bug: Shift click only works if the current selection was selected with a simple left click. So shift clicking on different items to adjust the selection range won't work after the first one.

Right click on a feature or category brings up a context menu. The context menu is described on page 47.

Double click has no function the `Scene View` panel.

Other controls are summarized in the Controls for Hierarchical Lists section above.

Introduction to Features

This book groups features into three types:

- A zone, model, plant, reference, truck, sound, or sounddomain is a “simple object”. It has a single location and orientation (even if it has no visual presence).
- An overlay or river is a “spline object”. It is a curve that links a number of points, each with a position but not an orientation.
- Everything else is just a feature, not an object. This includes geometry features, PbrMaterials, and distributions.

Simple objects and spline objects are together referred to simply as objects.

Each feature may have one or more named properties and/or one or more associated bitmaps.

A **named property** has a name and a value, and it is displayed in the properties panel at the bottom of the `Scene View`. Named properties are recorded in your map's XML file in the prebuild directory. Unless otherwise stated, references to “properties” in this guide refer to named properties. Named properties can be edited in the property panel, described below.

A **bitmap** is painted onto the terrain using a brush. Each pixel in the bitmap corresponds to a local region of the terrain and is displayed in the corresponding section of the main panel (which itself has many pixels on the screen). Each bitmap is recorded in a file in your map's directory within the prebuild directory. The name of the bitmap file is implicit for some features, and it is a named property for other features. Using brushes is described in more detail in TBD.

Property Panel

At the bottom of the `Scene View` panel is an unlabeled sub-panel that I call the property panel. It shows the named properties of the item selected in the main `Scene View` panel and allows those properties to be edited where applicable.

Most named properties can be edited directly in the properties panel, either by directly typing a new value or by some other means of selecting a new value. The editing conventions are different depending on the property type. Editing methods for each property type are described in the sections below.

When a property is edited, the changed values are displayed with bold text in the properties panel. The text remains bold until you change the selection or until you change a value back to its previous value.

If the changed values result in a visual change to the feature, it is immediately redrawn with its new appearance.

Group	MainGroups
Brand	air_conditioner_01
Tag	
Collision Type	STATIC
Position	
X	118.876549
Y	20.078430
Z	-74.301620
Rotation	
X	0.000000
Y	-0.000000
Z	-0.000000
Scale	2.000000
Disable Day Static Shadow	True
Disable Night Static Shadow	True
Freeze Physics	True
Animation Camera Frame Name	<No anim camera frames with...

A simple object's position, orientation, and scale properties or a spline object's point positions can also be modified by manipulating the object's interface in the main panel. The new values are updated in the property panel when you release the mouse button, but unlike manually entered values, the new values are not displayed in bold.

When selecting named properties, the **Tab** key switches between the left column to the right column, and the **Enter** key returns to the left column. If the selection is at the top of a set of related properties (e.g. **Position** or **Rotation** in the figure above), the **Enter** key expands or collapses the set.

Bug: For an editable text value, pressing the **Tab** key confirms the change (and updates the main panel if appropriate), but it does not return to the left column. Instead, neither column is selected. Rather than use the **Tab** key, I recommend that you use the **Enter** key to confirm the change and return to the left column.

A few named properties are for display only or otherwise cannot be edited once the feature is created (e.g. the map size).

Other controls for navigating through the list of properties are summarized in the Controls for Hierarchical Lists section on page 37.

Edit a Property by Typing Directly

For some properties, you can directly type a new value. Click in either the left column (name) or right column (value) of the desired property in the property panel and begin typing. If you clicked in the left column, your new text entirely replaces the old value. If you clicked in the right column, your new text is inserted where you clicked, and you can generally edit the old or new text. Pressing tab or enter or left clicking somewhere else completes the data entry.

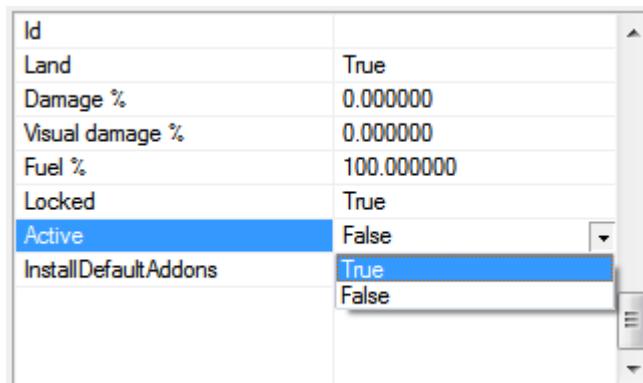
Group	MainGroups
Org X	118.834381
Org Z	-72.565430
Dir	(1; 0)
Id	I am typing directly...
Icon 30x30	
Icon 40x40	
Name	

Many named properties will look like they have a value that you can type directly, but when you click the property some other interface pops up. These other interfaces are described below.

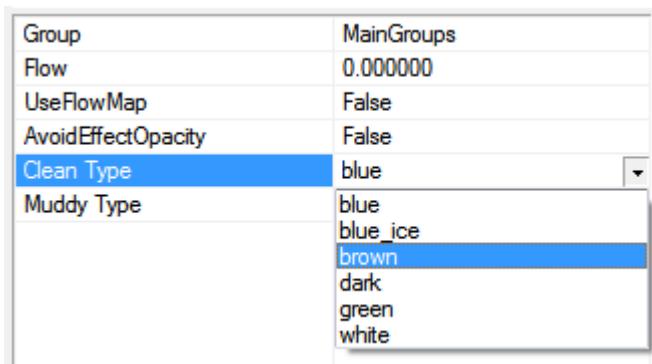
Edit a Property Using a Dropdown Menu

For a property with a dropdown menu, a small downward-pointing arrow appears when you select the property. Click the arrow to get a menu of values that you can choose among. You can also click at the right end of the property where the arrow will appear to select the property and invoke the dropdown menu with one click.

Properties that can be either True or False are the most common use of the dropdown menu.



When the property is selected, you can also type the first letter of the desired value to immediately select that value. If there are multiple values that start with the same letter, typing the letter cycles among those values. Picking a dropdown value by typing is not sensitive to case, so e.g. you can press **t** for **True** and **f** for **False**.



Finally, you can double-click the property to cycle among the values. This is particularly handy for switching between True and False. You have to select the property before you double click in the right column. You can double-click the left column without selecting the property first.

(Now aren't you glad you're reading this book? That's the kind of secret knowledge that could shave **seconds** from your map editing time.)

Edit a Property Using a Slider

For a property with a defined range of numerical values, clicking in the right column of the property causes a slider to appear to the right of the property value. The slider can be manipulated using the mouse or keyboard:

- Drag the slider indicator to the right or left to quickly increase or decrease the value.
- Click on the slider to the right of the indicator or press the **Page Down** key to increase the value by a large step.
- Click on the slider to the left of the indicator or press the **Page Up** key to decrease the value by a large step.
- Press the right arrow key to increase the value by a small step.
- Press the left arrow key to increase the value by a small step.
- Press the **End** key to set the value to the maximum value.
- Press the **Home** key to set the value to the minimum value.

There is no way to type a value directly into a slider property.

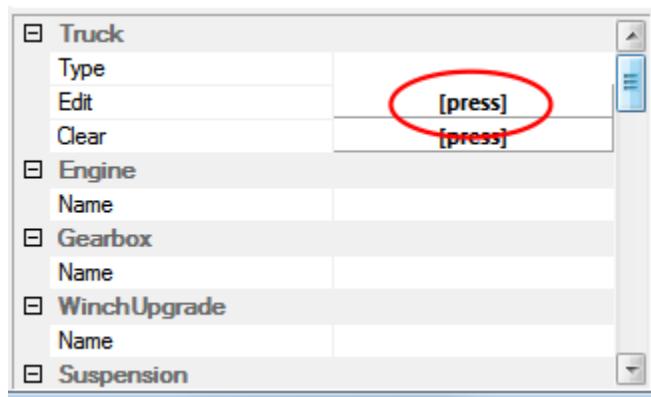
Id	
Land	True
Damage %	50.000
Visual damage %	0.000000

Edit a Property That Selects an Asset

For properties that chose among things that can be represented visually, the editor uses a **Select Asset** window.



For many objects, the **Select Asset** window pops up when the object is created in order to choose its type. For other objects, an asset can be selected at any time. For these, the editor displays the name of the current asset as an uneditable value, but it adds a second pseudo-property that allows you to edit the asset property. The name of the pseudo-property is **Edit**, and its value displays as **[press]**. Left click the **[press]** text to open the **Select Asset** window. (The text acts like a button, even though it doesn't look like one.)



Bug: The first time you open a **Select Asset** window for a particular asset type, the editor may lock up for 5 or more seconds as it creates icons for all possible assets of that type. It will be faster for subsequent asset selections of that type.

Bug: In most programs, when you click in the scrollbar above or below the current scroll position, the window scrolls up or down by one page. However, in the **Select Asset** window, it instead scrolls by a fixed percentage of the total number of entries. This makes it hard for your eye to predict where to look for assets that have newly scrolled into view. And for assets such as truck add-ons, the fixed scroll percentage scrolls so far that it actually skips over many of the available assets. When you click the up or down arrow buttons at the ends of the scrollbar, it scrolls by a smaller fixed percentage. This has a similar bug, but at least it scrolls by a small enough percentage that it shouldn't skip over any assets (unless you've added enough custom assets to greatly expand the list).

To select an asset, left click it in the **Select Asset** window and then click **OK** or press enter. Or double-click an asset to choose it and immediately confirm your choice.

Click **Cancel** or press the escape key to cancel the asset selection. If the **Select Asset** window was opened during the creation of a new object, the object creation is canceled. If you click **OK** or press enter when no asset is selected, it is the same as clicking **Cancel**.

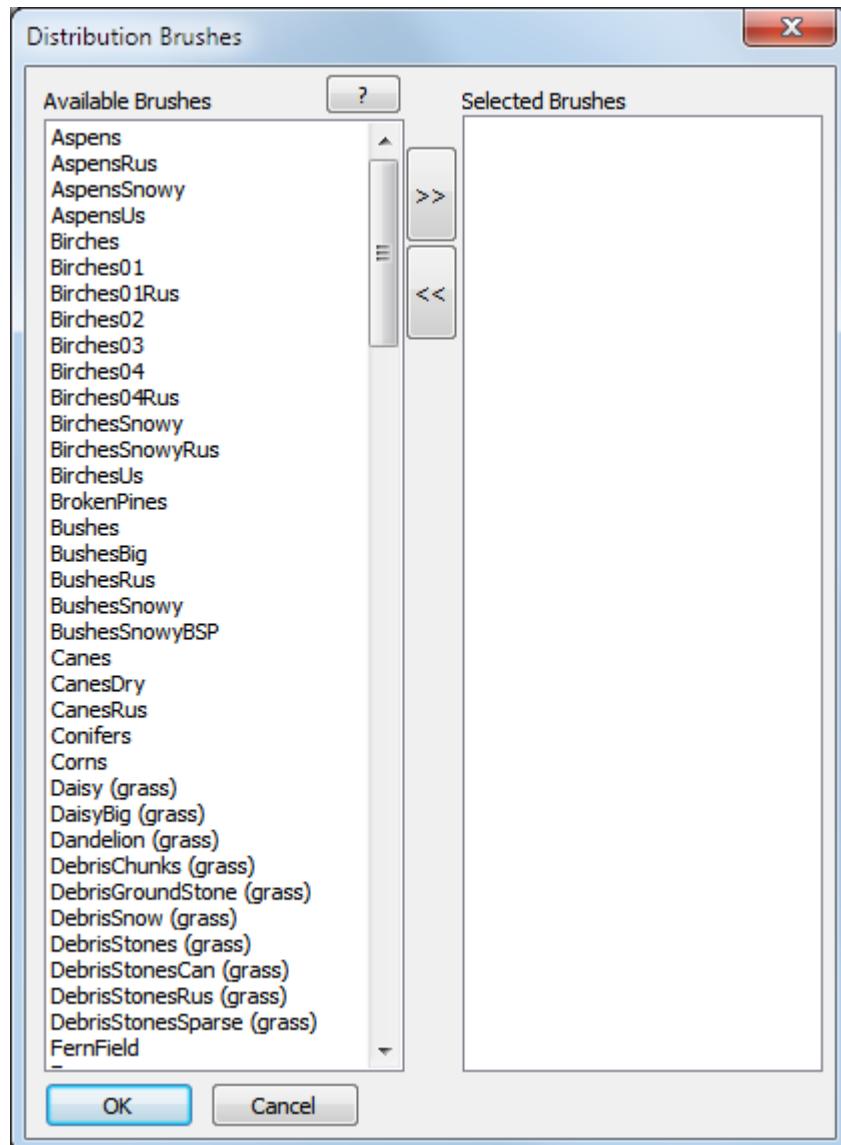
For properties where an asset is not required, you can click **[press]** next to the **Clear** pseudo-property in order to clear the selected asset.

Bug: The `Clear` property is shown for many properties where an asset is required. For these properties, clicking [press] next to `Clear` does nothing.

You can filter the list of assets by typing in the blank panel near the top of the dialog box. Only assets that include your text somewhere in the name are shown in the dialog box.

Edit a Property that Selects Brushes

The appearance of some features depends on which brushes are attached to the feature. Zero, one, or many brushes can be selected using the Select Brush window. The editor displays the names of the current brushes as the value of a `List` property which can be modified using the following `Edit` pseudo-property. The value for the pseudo-property displays as [press]. Left click the [press] text to open the brush selection window.



Move a brush from the left **Available Brushes** column to the right **Selected Brushes** column by clicking its name and then the **>>** button. Move it back by selecting it from the right column and clicking **<<**. Or quickly change a brush from one side to the other by double clicking it in either column.

When you are done, click **OK** or press enter to confirm your selection of brushes. Or click **Cancel** or press the escape key to cancel your changes.

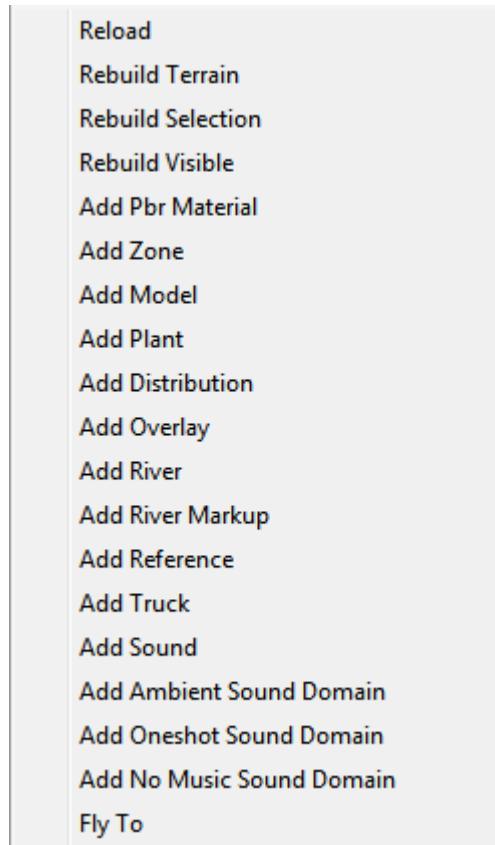
The Context Menu

The SnowRunner Editor has a context menu that can be invoked by right clicking in a couple of different places. The context menu always includes a number of common entries, and more entries are added for the context of the chosen feature.

In most cases, if you right click anywhere in the main panel, the context menu appears for the currently selected feature. However, if a feature with a bitmap property is selected, right click in the main panel operates the paint brush.

If you right click on a feature in the **Scene View** panel, the context menu appears for that feature. If you close the context menu without selecting an item from the menu, the feature is selected as if you left-clicked on it.

The rest of this section describes the common context menu entries, plus a few entries that are used by a number of features.



Bug: The context menu only works in the main panel when something is selected at the `(Terrain)` level or below. Right click in the main panel is ignored if the top-level `Scene` is selected or if nothing in the `Scene View` panel is selected (so the property panel is blank). This most commonly occurs after the map is saved, after the terrain is rebuilt, or after a feature is deleted. My habit is that if right click doesn't work in the main panel, I just left click and then right click again.

Reload

`Reload` discards all unsaved changes and reloads the map from disk. The editor asks for confirmation before reloading.

Rebuild Terrain

`Rebuild Terrain` redraws everything in the map. For a large, complex map, this can take a very long time.

If the editor has taken shortcuts in its iterative drawing (such as not redrawing shadows), `Rebuild Terrain` will correct the view. It also occasionally reshapes some features such as terrain height and rivers that need a wider context to correctly join together.

Each of the `Rebuild` actions first commits any property edits in progress, and it clears any current feature selection.

Rebuild Selection

`Rebuild Selection` redraws everything in the selected terrain blocks. This takes some shortcuts, but not as many as the editor takes in its iterative drawing. Because selecting multiple terrain blocks is a pain, I find this option to be not very useful.

Rebuild Visible

`Rebuild Visible` redraws all visible terrain blocks. This takes some shortcuts, but not as many as the editor takes in its iterative drawing. This is occasionally useful for updating the view of certain features, although I can never remember what. I'll sometimes try it, and then only choose the slow `Rebuild Terrain` option if necessary.

Add <Feature>

`Add <Feature>` adds the named feature to the map and selects it. The feature's position is initialized to the terrain in the center of the main panel. If there is no terrain visible at that spot, the feature is added at a corresponding point off the map.

If a map feature is selected, the feature's category will often add a context menu item such as "Trucks - Add Truck". This is the same as the generic "Add Truck" menu item, but is a little closer to the mouse when the context menu is invoked.

Fly To

`Fly To` moves the camera (without rotation) to a position in which all selected terrain blocks are visible. This option is only available when a feature is not selected.

<Feature> - Fly To

`<Feature> - Fly To` moves the camera (without rotation) to point at the selected feature. If multiple features are selected, the menu item is `Selected - Fly To`, and it moves the camera such that all selected features are visible. If the selected feature is not an object, `<Feature> - Fly To` moves the camera to the selected terrain block like `Fly To` above.

You can also double click a model or plant in the main panel to select and fly to it.

<Feature> - Delete

`<Feature> - Delete` deletes the chosen feature and clears the current selection. The secret keyboard shortcut is the `Delete` key.

If multiple features are selected, the menu item is `Selected - Delete All`, and it deletes all of the selected features.

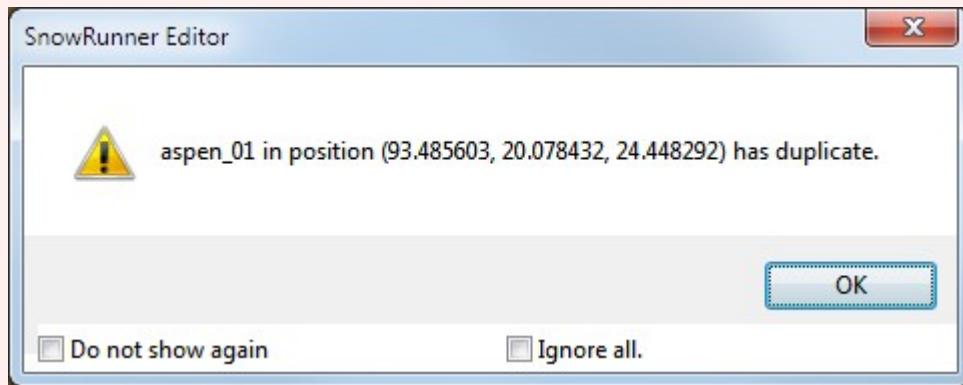
There is a secret undo function for delete. Pressing `Ctrl-Z` restores the deleted feature. `Ctrl-Y` redo what was undone, so in this case it re-deletes the feature. There is only one level of undo.

Bug: `Ctrl-Z` restores only the position and rotation and a few other properties of the deleted feature. Most properties of the feature are lost. E.g. model properties are restored perfectly, but all useful truck properties are lost.

<Object> - Duplicate

`<Object> - Duplicate` makes a copy of an object and selects the copy. This copies all of object's properties to the new object and selects it. The object can also be duplicated with a shortcut key, `ctrl-D`.

Bug: Duplicating a model or plant pops up a warning that the object is duplicated. This dialog may be repeated even when you delete the duplicate object. The checkbox for `Do not show again` only works if the next warning dialog has the exact same text, so it won't suppress the warning when duplicating a different object.



You might want to cultivate a habit of pressing `Ctrl-D` to duplicate and then `Enter` to dismiss the dialog for models and plants. Note, however, that the warning dialog does not appear for other object types.

If multiple features are selected, the menu item is `Selected - Duplicate All`, and it deletes all of the selected features.

Bug: Duplicating multiple features at once may cause an extra one to spawn every time the duplicate warning dialog pops up with no obvious way to make it stop.

<Category> - Add <Feature>

`<Category> - Add <Feature>` is the same as the generic `Add <Feature>` menu item, but is a little closer to the mouse when the context menu is invoked on another feature of the same type.

If the selection is within a group, there is also a menu item for `<group name> - Add <Feature>`, and it adds the new feature directly to the group.

<Category> - Add Group

`<Category> - Add Group` creates a new hierarchical container under the feature category. Groups allow some amount of organization in maps which may contain a very large number of features.

Groups can only be created directly within a category. They cannot span categories, and they cannot be nested.

A new group is assigned a random numeric ID. This can be changed to any desired name.

Features within a group are recorded in a separate XML file which is saved in the map's `subgroups` directory. This XML file can be copied into another map's `subgroups` directory to effectively copy all of the grouped features to that map. Their locations from the original map may not be appropriate for the new map, but it is easy to select all of the grouped features in the new map and move them together.

<Category> - Collapse All

`<Category> - Collapse All` contracts the category and all groups within the category so that their contents are hidden from the list.

If the selection is within a group, there is also a menu item for `<group name> - Collapse All`, and it contracts only that group.

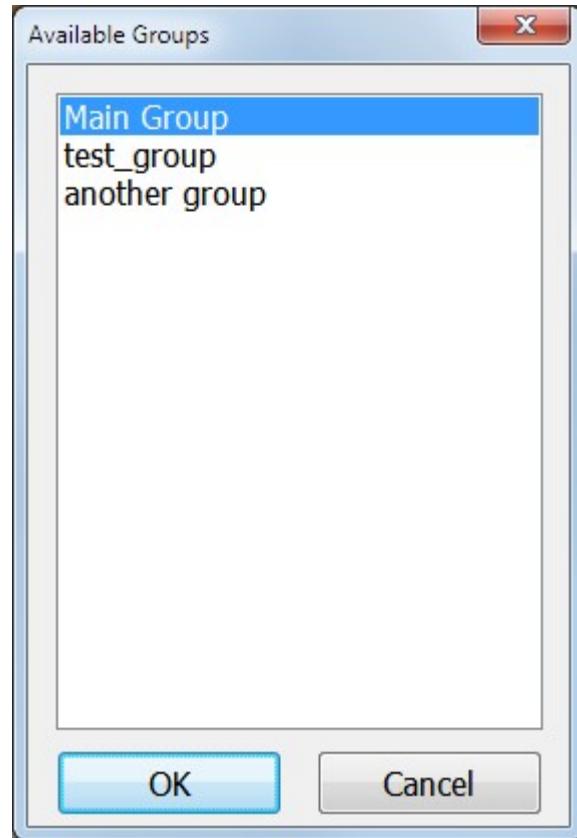
<Category> - Expand All

`<Category> - Expand All` expands the category and all groups within the category so that their contents are shown in the list.

If the selection is within a group, there is also a menu item for `<group name> - Expand All`, and it expands only that group.

<Category> - Move to other group

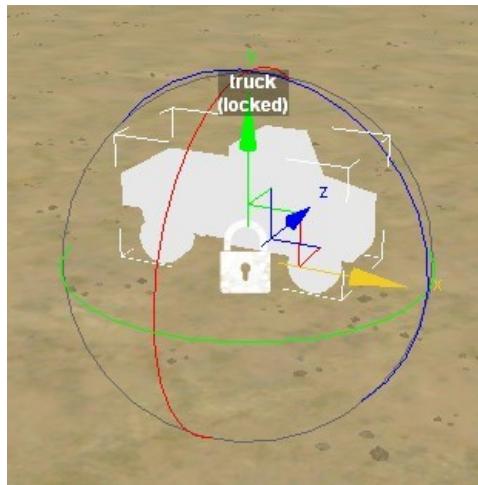
<Category> - Move to other group or <Group> - Move to other group pops up a dialog to allow the selected feature(s) to be moved to another group or back into the main category list (called Main Group in the dialog).



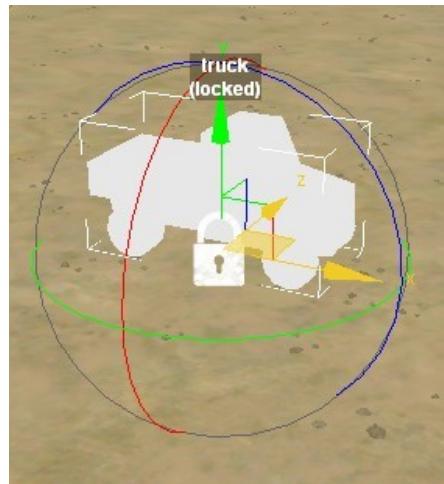
Move a Truck, Model, or Plant

Manipulate an Object in the Main Panel

When an object is selected, the main panel draws on the object a set of colored arrows corresponding to the X/Y/Z coordinate axes. Left click and drag an arrow to move the object back and forth along that axis. By default, a truck or plant is attached to the ground and cannot be raised into the air, so dragging the green Y arrow does nothing. A model can be raised or lowered freely.

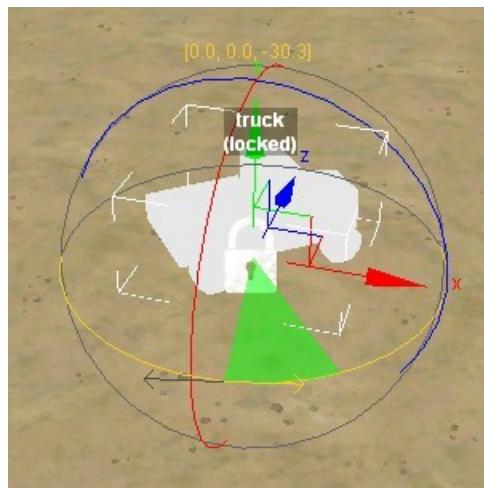


Connecting the bases of the arrows are three squares in 3-D space that designate the XZ plane (ground plane), XY plane, and YZ plane. Dragging one of these squares moves the object around in the 2-D space associated with the plane. The squares will often partially overlap on the screen, which can make it difficult to grab the right one. However, moving in the ground plane is the most generally useful, and the editor always prioritizes that one when there's an overlap. Because a truck or plant is attached to the ground by default, dragging the XY plane or the YZ plane will only move the truck in the X or Z directions, respectively.



The object is also surrounded by a sphere composed of three colored circles. Left click and drag the green circle to rotate the object around its vertical axis (the Y axis). By default, a truck is attached to the ground and cannot tilt away from it, so the red and blue circles do nothing. Although the base of a plant is attached to the ground, it is still allowed to tilt. The red and blue circles rotate the plant around the X and Z axes, respectively. A model can also be freely rotated in the same manner.

When dragging the mouse to rotate an object, your intuition is likely to drag the mouse in a circle, but that is not how the editor works. When you start dragging, the editor draws an opposing pair of arrows tangent to the circle at the location where you first clicked. Drag the mouse linearly in the direction of one of the arrows to rotate the truck. The more you drag, the more it rotates (even past 360°), although rotation stops when the mouse leaves the main panel.



The object is surrounded by a virtual box that designates the boundaries of the truck in the X, Y, and Z axes. The corners of this box are shown with white lines. Because the box has a fixed orientation, its boundaries expand and contract to follow the corners of the object as it rotates.

Manipulate an Object in the Scene View

An objects position and orientation can also be edited in the property panel at the bottom of the [Scene View](#).

The position of an object is listed in the [Position](#) group with three properties [X](#), [Y](#), and [Z](#). The values can be edited directly and are measured in meters.

Position	
X	-27.393368
Y	20.078434
Z	-31.153299
Rotation	
X	-0.000000
Y	-0.000000
Z	-0.000000

The orientation of an object is listed in the **Rotation** group. The **X**, **Y**, and **Z** properties specify the objects rotation around each axis, measured in degrees.

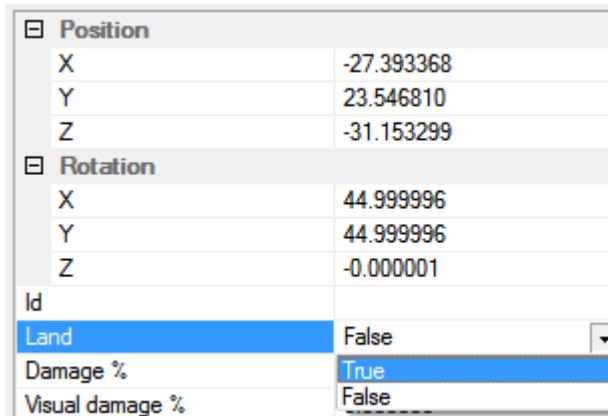
Bug: If you edit the position or rotation properties of an object by hand, those changes will not be saved. The object will revert to its previous position and orientation if you pack the map or reload the map. When hand editing an object's position or rotation, also edit at least one other property, even if you change it right back to its original value. Or use the widgets in the main panel to move and rotate the object. Changes made by these widgets will save correctly.

For any given orientation, there are many combinations of **X**, **Y**, and **Z** that achieve the same rotation. You might notice that when you click away from the object and then re-select it, the Editor has changed the rotation property values. However, they still result in the same orientation.

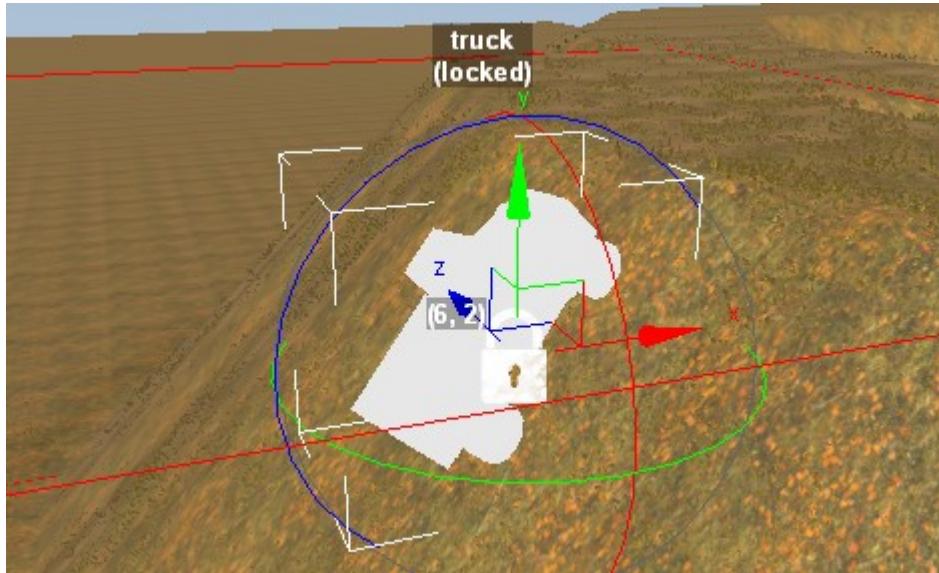
The coordinate systems for position and rotation are described in more detail in the appendices of this guide.

Land and Do Land

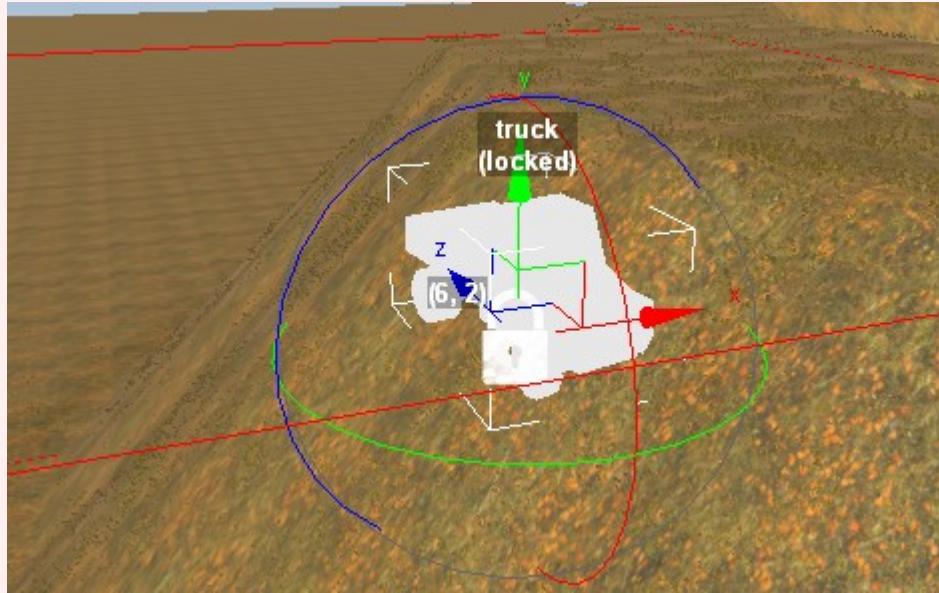
The other property of interest for moving and rotating trucks is the **Land** property, which can be set as True or False. The equivalent of “Land” for plants is “Do Land”.



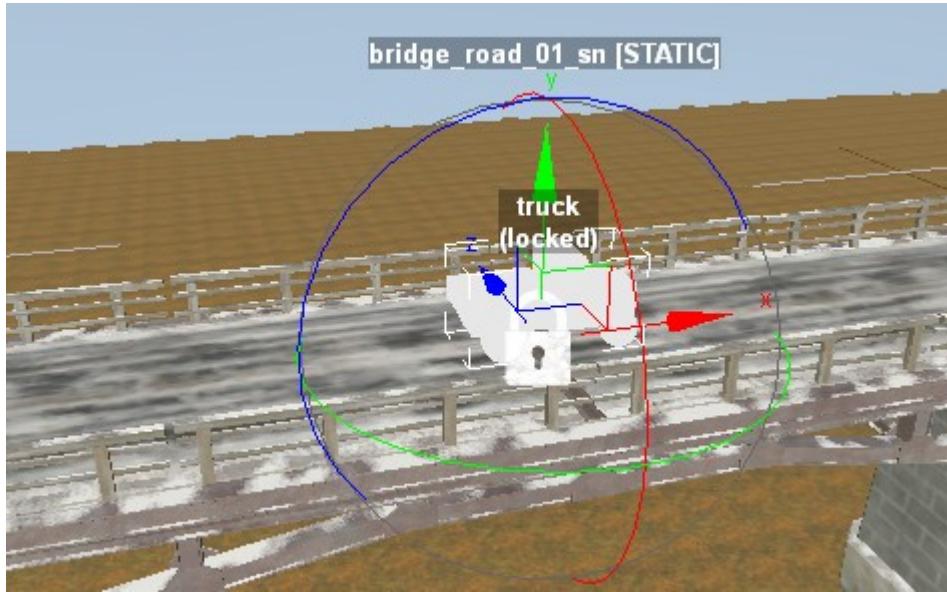
When the Land or Do Land property is True, the truck or plant is attached to the ground. A plant keeps its orientation when it is attached to the ground. For a truck, the Editor tilts the truck so that its wheels make maximum contact with the ground.



Bug: The Editor shows the truck pitched correctly up and down, it does not show it tilted correctly side to side. However, its initial position will be tilted correctly in the game.



When the Land or Do Land property is False, the truck or plant can be moved freely. It can be raised into the air, buried in the ground, and tilted to any orientation. In most cases, this will look unnatural and result in strange physics. But it can also be useful, e.g. to place a truck on a bridge.



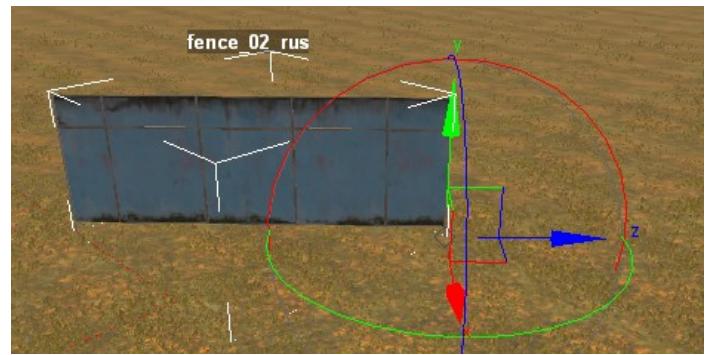
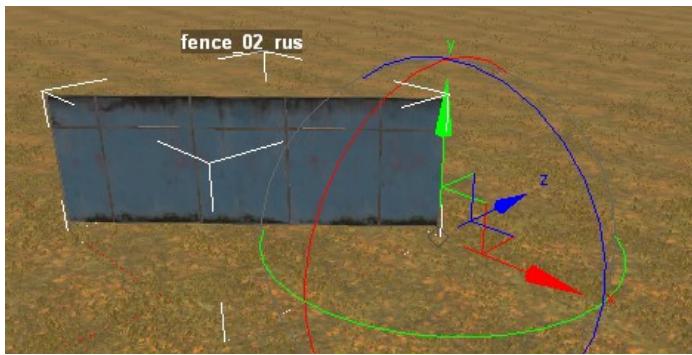
Models don't have a Land or Do Land property, but they have an additional context menu item for "Do Land". When this is selected, the editor moves the model down until its base point (which may or may not be the bottom of the model) touches the ground. The model is not attached to the ground, however, and can still be freely manipulated from its new location.

When a truck is attached to ground, it orients itself for maximum contact. When a plant is attached to ground, its narrow base means that it makes good contact regardless of orientation. However, it can be difficult to cleanly place a model on uneven ground. Be sure to carefully check around the base of the model for gaps and either re-orient the model or adjust the ground height until it has a solid connection.



Local Transform

When moving and rotating objects, it can be helpful to manipulate them using their local axes rather than the global axes. The **Local Transform** option in the toolbar can be toggled by clicking it or pressing **ctrl-L**. Using local axes can make tilting more intuitive, and it can make it easier to line up repeated objects such as fences.



Edit Other Truck Properties

The previous sections described the steps to create and move generic objects. This section describes how to edit the properties specific to trucks.

Select Truck Components

Choose the Truck Type

The primary property for any truck is its type. When a truck is selected, this property is the top one in the property panel. To select the truck type, left click [press] for the `Edit` pseudo-property in the `Truck` group. This brings up a list of trucks from which you can choose your truck type.

Bug: The first time you click here, the editor may lock up for 5 or more seconds as it creates icons for all possible trucks. It will be faster for subsequent uses.

Bug: The `Truck` group includes a `Clear` pseudo-property, but clicking [press] next to `Clear` does nothing.

Once a truck type has been selected, it appears in the properties panel, and the ghost view of the truck in the main view changes to reflect the selected truck type.

Bug: Although the truck silhouette updates immediately, the name of the truck type isn't displayed correctly in the main panel until you deselect the truck.

Note that the truck list includes trucks from all possible DLC even if you do not own that DLC. If a player tries to play a custom map with one of these trucks without owning the corresponding DLC, the truck will not appear. Keep this in mind when selecting trucks for your map.

You can choose a different truck type by clicking next to `Edit` again and choosing a new truck. Choosing a new truck type discards the previous selection, but keeps all other properties including trailers and addons.

The Editor uses lowercase truck names that differ slightly from the names used in the game, but it's generally obvious how the names in the Editor correspond with the names in the game.

In the truck selection dialog you can choose to select a trailer instead of a truck. If you select a trailer this way (instead of the usual trailer selection below), the trailer is parked by itself as if it is a truck. Cargo can be loaded on this trailer as described in the Choose Add-ons section below.

By the way, because wheels are a separate asset that are attached to the truck/trailer by the game, they are not shown in the images in the asset selection dialog. They also aren't shown on the silhouette of a truck in the main panel, although they are shown for trailers.

Identify Compatible Truck Parts in the Virtual Garage

Since each truck generally has many available parts, few of which are shared with other trucks, there are a huge number of part names to keep track of. Since each truck can only install a subset of parts, and only in certain combinations, the best way to find compatible truck parts is in the game itself.

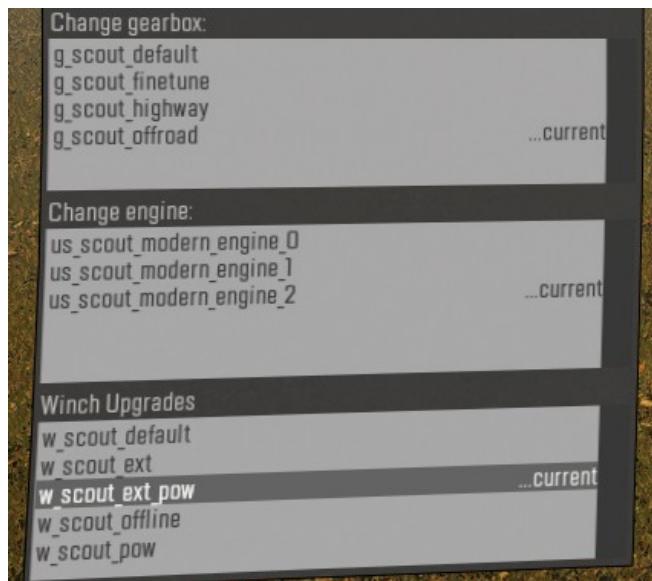
The various truck parts and trailers have lowercase names in the Editor that differ from the names used in the normal game. However, when playing in your own custom map or in any proving grounds map, the **Garage** option opens a virtual garage that allows you to easily discover the lowercase names of compatible parts.

When selecting a truck part or attached trailer, the Editor does not check that it is compatible with the truck or the other parts. If it is not compatible, it will simply be ignored by the game.

For more information on the dev tools and the virtual garage, see the Dev Tools: Virtual Garage section on page 151.

Choose an Engine, Gearbox, Winch, and Suspension

Each of these property sections has a single **Name** property with a value that is initially blank. Enter the name of a compatible truck part into the corresponding property value for that part to be installed in the truck. If the name is left blank (or an incompatible name is entered), the default for the truck is used instead.



Engine	us_scout_modern_en...
Gearbox	g_scout_offroad
WinchUpgrade	w_scout_ext_pow

Choose Wheels

The **Wheels** section has properties for **Type**, **Rim**, **Tire**, and **Scale**. All of these can be determined from the tire and rim names shown in the virtual garage.

- The **Type** value is the text prior to the **/** in the name of the tire and rim. (It will always be the same for both.)
- The **Rim** value is the text after the **/** and before the **x** in the name of the rim.
- The **Tire** value is the text after the **/** and before the **x** in the name of the tire.
- The **Scale** value is the text after the **x** in the name of the tire and rim. (It will always be the same for both.)

If any name is left blank (or an incompatible name is entered), a compatible default is used.



Choose a Color

The **Customization** section has a property for **PresetId**, which sets the color scheme for the truck. The preset ID is a number that can be found in the **Customization** section of the virtual garage following the word **Preset**.

For the default color scheme, enter the value **-1**. Note that this is different than preset ID **0**.

The standard color schemes are numbered **0 – 31**. Many of these are a single solid color, but some are multi-toned. Experiment in the garage or virtual garage to find a pleasing scheme.

Special skins and vinyl wraps are numbered **32** and higher. Note that they are typically available only with DLC installed, and only for certain trucks.



Choose a Trailer

Selecting a trailer is performed much like selecting a truck, this time using the pseudo-properties in the [Trailer](#) section.

If you select a trailer that is not compatible with the truck type, the editor simply records the choice without any error notification. Make sure to find a compatible trailer in the virtual garage.

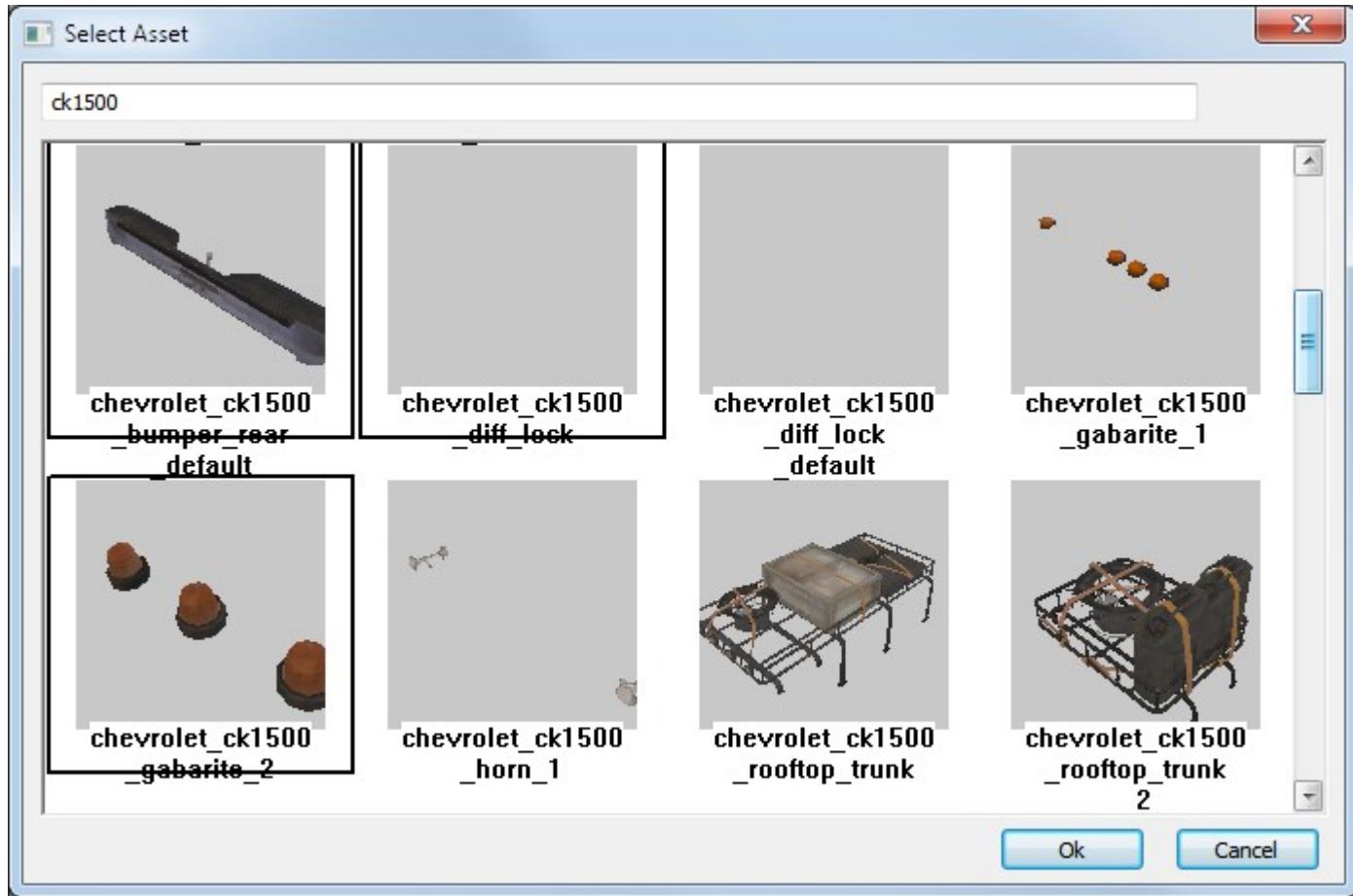
The trailer is not drawn in the main view. You will have to use your intuition about its length to ensure that it doesn't overlap any other objects.

Choose Add-ons

Selecting add-ons is performed using the pseudo-properties in the [Addons](#) section. Unlike for trucks and trailers, however, this dialog box allows you to select multiple items before clicking [OK](#). Clicking an add-on in the dialog once selects it. Clicking it again deselects it. Selected add-ons are indicated with a black border.

Bug: The first time you click here, the editor may lock up for 30 or more seconds as it creates icons for all of the many add-ons. It will be faster for subsequent uses.

You can filter the list of add-ons by typing in the blank panel near the top of the dialog box. Only add-ons that include your text somewhere in the name are shown in the dialog box.



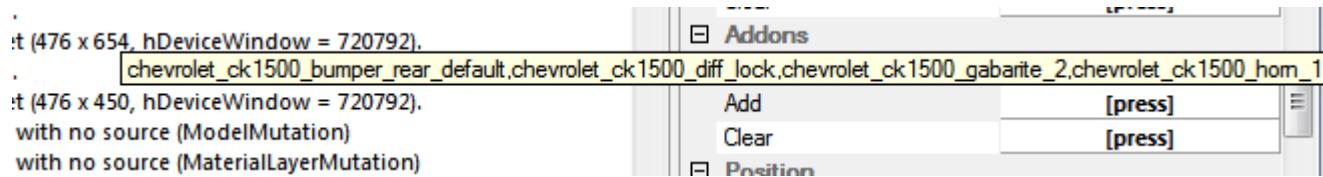
You may have noticed that the pseudo-property name for add-ons is **Add** instead of **Edit**. Once some add-ons are selected, click **[press]** next to **Add** again to add more items to the list.

Unfortunately, there is no way to selectively remove add-ons from the list. Instead, click **[press]** next to **Clear**, then re-add the desired add-ons.

As usual, the Editor does not check the compatibility of add-ons. Use the virtual garage to find the names of add-ons that are compatible with the truck and can be used together. The order of add-ons doesn't matter, as long as the necessary prerequisites are somewhere in the list.

Add-ons are not drawn on the truck in the main panel.

The list of addons may be longer than will fit in the properties panel. Hover your mouse over the list of addons to get the full list in hover text.



Default Add-ons

Some add-ons have a name ending with `_default`. If a conflicting add-on is not added to the property value, the game automatically installs the default.

At the bottom of the property list is a property called `InstallDefaultAddons`. As far as I can tell, this property does nothing. It is possible to uninstall default add-ons in the virtual garage (e.g. to leave a truck without any exhaust pipes), but the game always installs (or re-installs) the necessary defaults when the map is loaded.

Cargo

Cargo can be loaded onto the truck as an "add-on". All cargo add-ons can be found in the addon dialog by filtering for the word "cargo". Note that in a few cases (particularly for logs), the cargo is shaped differently for a particular truck and so has a unique name. Be sure to pick the right add-on for your truck. Unfortunately, the dev tools cannot help us test the compatibility of cargo.

Warning: Unlike other add-ons where order doesn't matter, the appropriate truck bed for the cargo must appear in the add-on list before the cargo. This means that typically you will add the truck bed first, confirm and close the dialog box, and then re-open the dialog box to add the cargo.

To load multiple copies of the same cargo on a truck, add one, exit the dialog, then add another. Each copy is added to the list of add-ons, and all the cargo that fits will be loaded onto the truck. In the case of cargo of different types, cargo earlier in the list is loaded onto the truck closer to the front.

Truck ID

The `Id` property is optional. If filled, it should provide a unique identifier for this exact truck (i.e. not another one of the same type). This ID allows the truck to be linked to an objective (page 124), e.g. as a requirement or a reward.

For safety, I recommend that the localization ID be limited to a combination of lowercase `a – z`, uppercase `A – Z`, `0 – 9`, and `_`. It should not include other characters, including spaces.

Starting Damage and Fuel

The amount of damage and fuel that a truck starts with are set by properties.

The `Damage %` property specifies the initial damage to the truck's components. Each component is damaged by the same percentage including wheels. The default is 0%, no damage.

The `Visual damage %` property specifies the initial damage to the truck's body panels. Damage is evenly distributed across the truck. Each component is damaged by the same percentage, including wheels. The default is 0%, no visual damage. Even 5% visual damage can look rather significant (shown below), so don't overdo it.



The `Fuel %` property specifies the initial fuel in the truck. The default is 100%, fully fueled.

Any repair or refueling add-ons always start out full and are unaffected by the above properties.

If a trailer is placed on its own, the `Fuel %` property specifies the initial level of any refueling tank on the trailer. The `Visual damage %` property specifies the initial damage to the trailer's appearance. The `Damage %` property has no effect on a trailer since it has no components that can take damage. A repair trailer always starts with a full load of repair parts.

Locked and Unlocked Vehicles

The `Locked` property determines whether a truck or trailer is locked at the start of the map. When the `Locked` property is `True`, the Editor adds a `(locked)` suffix to its name and draws a padlock icon on its silhouette in the main panel. A truck starts locked by default.

If the player drives very close to a locked truck or trailer, it will be “discovered”.

Regardless of its locked or unlocked status, the player cannot drive a truck until it is at least revealed on the map. For an unlocked truck

The player can discover an unlocked truck by revealing it on the map. This can be done by driving somewhat near the unlocked truck or by activating a watchtower that reveals the truck. See the Discovery of Vehicles section on page 150 for more information about the discovery process.

A locked truck, however, remains locked even after it is revealed on the map. The player can only activate a locked truck by driving very near it.

Reserved Trucks

A truck (or trailer? TBD) can be reserved for use in a truck delivery goal (page 142) or truck repair goal (page 143). If so, its locked/unlocked status is ignored. A player cannot “discover” a reserved truck, nor can she enter the truck or otherwise drive it. An objective may reserve a truck permanently, or the truck may be awarded to the player upon completion of an objective.

The Active Truck

The `Active` property determines which truck the player starts in. If you set the `Active` property to `True` for a truck, the Editor automatically sets all other trucks to `False` so that there is only one active truck. When the `Active` property is `True`, the Editor adds an `(active)` suffix to its name and draws a giant arrow pointing down at it. The `Locked` property is ignored when the `Active` property is `True`.

Bug: When an active truck is selected, the Editor draws a 32-meter gray disk around it in the main panel. This is a holdover from the MudRunner Editor, and it has no useful meaning in SnowRunner.

Although the Editor attempts to allow only one truck to be active, it is possible to end up with multiple active trucks, e.g. by duplicating an active truck. In this case, the game uses only the last truck marked active in the list of trucks in the `Scene View`, and it treats the other trucks as inactive.

If no truck is marked as active, then the player will spawn in free space with no ability to move the camera or enter a truck. If the `Tools` menu is enabled in the game, you can use it to jump into an existing truck. See the Dev Tools section on page 150 for details.

If a placed “truck” is actually just a trailer, the Active property has no effect in the game.

Hand Edit the XML

The properties for all of your trucks (and other data) is saved in your map's XML file in the prebuild directory. Editing the XML by hand is useful for tasks that are difficult or impossible to do in the SnowRunner Editor. For example, you can rearrange the trucks so that the active truck is first. Or you use find and replace to change every Locked property to False for testing purposes.

More detail about hand-editing the XML file appears in the appendices.

Terrain Height

Clicking any of the items under **(Geometry)** in the **Scene View** allows you to edit the terrain. Terrain features don't have named properties, and they are not saved in the map's XML file. Instead, the terrain is divided up into small squares, and data for each square is recorded as a pixel in a bitmap. The "brightness" for each pixel in the bitmap determines the strength of a corresponding feature in the terrain. For example, if the terrain height increases smoothly from minimum height on the left to full height on the right, it is recorded in the height bitmap as image that fades from black on the left to white on the right.

This concept of terrain features as bitmaps is reflected in the metaphor for editing those features: painting the terrain using a brush.

There are many features within the **(Geometry)** category, but this section is devoted to the **(Geometry)** feature itself.

Click **(Geometry)** under **(Terrain)** in the **Scene View** to edit the terrain height. A **Brush** dialog pops up that allows you to edit your brush parameters. Then draw on the map by holding down the right mouse button while moving the mouse (right-click and drag).

If you are satisfied with the results, left click in the main panel or on something else in the **Scene View** to deselect the **(Terrain)** tool. This commits your changes and writes them to the underlying bitmap file. (No separate save step is necessary for terrain changes.)

On the other hand, if you don't like your changes, click **Undo Current Changes** in the **Brush** dialog to revert your changes without updating the underlying bitmap file. Changes are reverted across all brush modes back to the point when you first entered the **(Geometry)** brush. You can then try drawing again or left click to exit drawing mode.

The **(Geometry)** brush has three modes: **Height**, **Flatten**, and **Smooth**. Each of these modes is described in the sections below.



Height Brush Mode

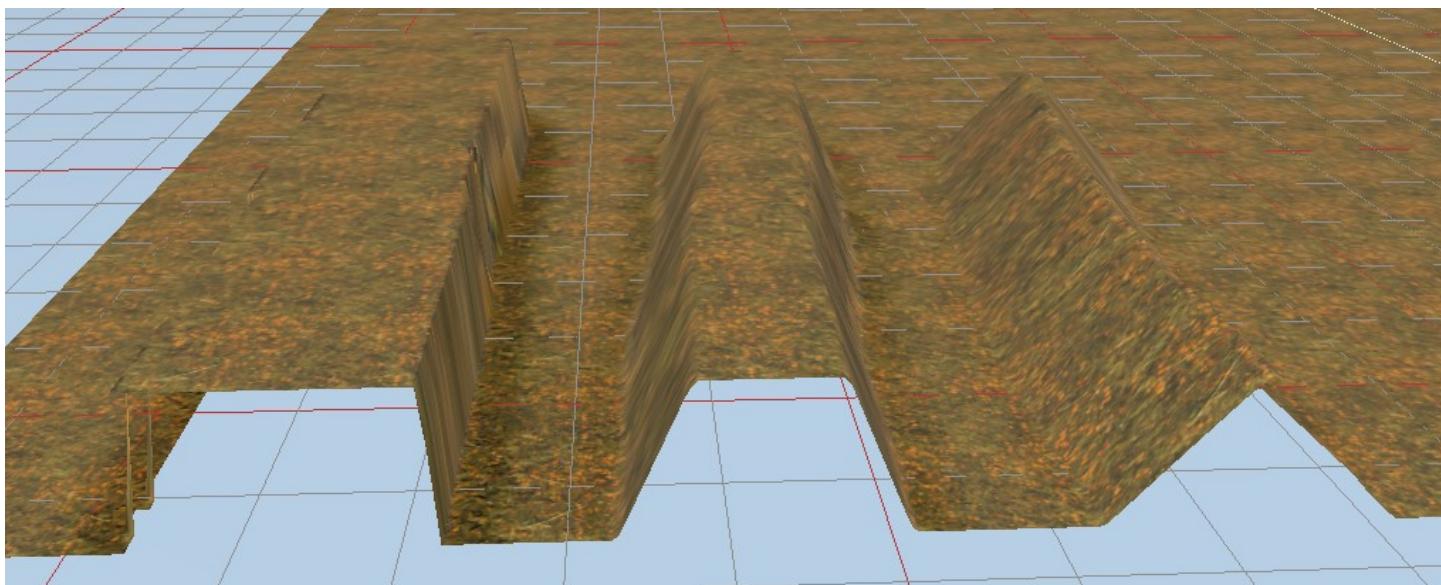
The default **Height** mode changes the terrain height directly.

The **Size** parameter in the **Brush** dialog sets the radius of the brush in meters. The slider is approximately logarithmic, so you have fine control of small brush sizes and coarse control of large brush sizes.

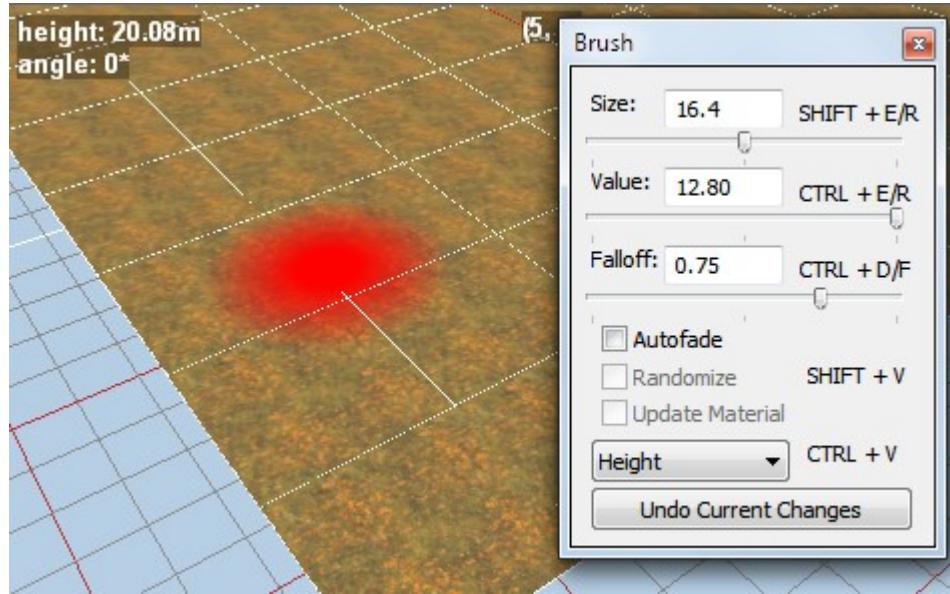
The **Value** parameter sets how much the terrain height changes at the center of the brush during a paint stroke. The value is measured in meters. Positive values increase the terrain height, and negative values decrease the terrain height.

The brush has an inner region in which the terrain height is changed by the full **Value** and an outer region where the change falls off to zero. The **Falloff** parameter determines the fraction of the brush radius that is dedicated to the outer brush region, where the change falls off. With a **Falloff** of 0.00, there is no outer region, and the entire brush paints the terrain with the full change set by **Value**. With a Falloff of 1.00, falloff is spread across the entire brush, so only the point at the center paints the terrain with the full change. In most cases, some intermediate value is desirable.

The figure below shows the effect of different **Falloff** values, with 0.00 on the left, 0.50 in the middle, and 1.00 on the right.



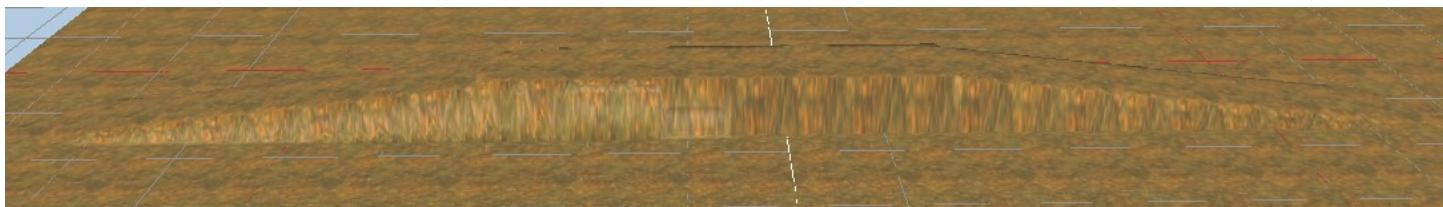
While in **Height** mode, a disc is drawn on the terrain under the mouse to preview the brush parameters. A red disc is drawn for a large positive **Value**, and a black disc is drawn for a very negative **Value**. The brush **Size** and **Falloff** are also represented on the disc. This provides a quick sanity check that your parameters make sense before you start to paint.



The disc is less strongly colored at reduced values. Unfortunately, when drawing with a small **Value**, the disc can be nearly invisible.

Autofade

The **Autofade** checkbox is mostly useful for drawing tire tracks in mud, not for drawing height, but you can experiment with it here. With **Autofade** enabled, when you right click and begin dragging in some direction, the editor adds a tail on your tint that fades in as if you had slowly lowered the brush onto the terrain while approaching the spot where you actually clicked. And when you let go of the right mouse button, the editor adds a tail that fades out as if you had slowly lifted the brush while continuing to drag in the most recent direction. The length of the tail is about 10x the width of the brush.



I find Autofade very difficult to control, but your milage may vary.

Autofade is available for ([Geometry](#)) when using the **Height** mode.

Flatten Brush Mode

Once you've painted some height changes onto the terrain, it is very difficult to get it flat again using the **Height** mode. To easily flatten terrain, change the **Brush** dialog to the **Flatten** mode. In this mode, the terrain at the point where you start your brush stroke determines the target height. Painting in this mode shifts the height of the painted terrain toward the target height.

The Editor keeps track of the most recent brush parameters you used in each mode. When you switch modes, the parameter values are reset to whatever you last used in that mode.

The **Size** parameter still has the same purpose; it sets the radius of the paint brush.

The **Value** parameter is now a value between 0 and 1, and it determines how quickly the painted terrain should change toward the target height. With a **Value** of 1.00, the terrain at the center of the brush jumps fully to the target height. With a lesser **Value**, the terrain under the brush shifts only the specified fraction toward the target height. But this is deceptive! For every pixel that you drag the brush, the Editor re-applies the smoothing effect across the entire brush. For example, with a **Value** of 0.50, as you start dragging the brush, the terrain near the center of the brush, shifts halfway to the target value, then shifts half of the remaining amount, etc. This is very different than the **Height** mode, where the brush effect is re-applied only if you start a new stroke.

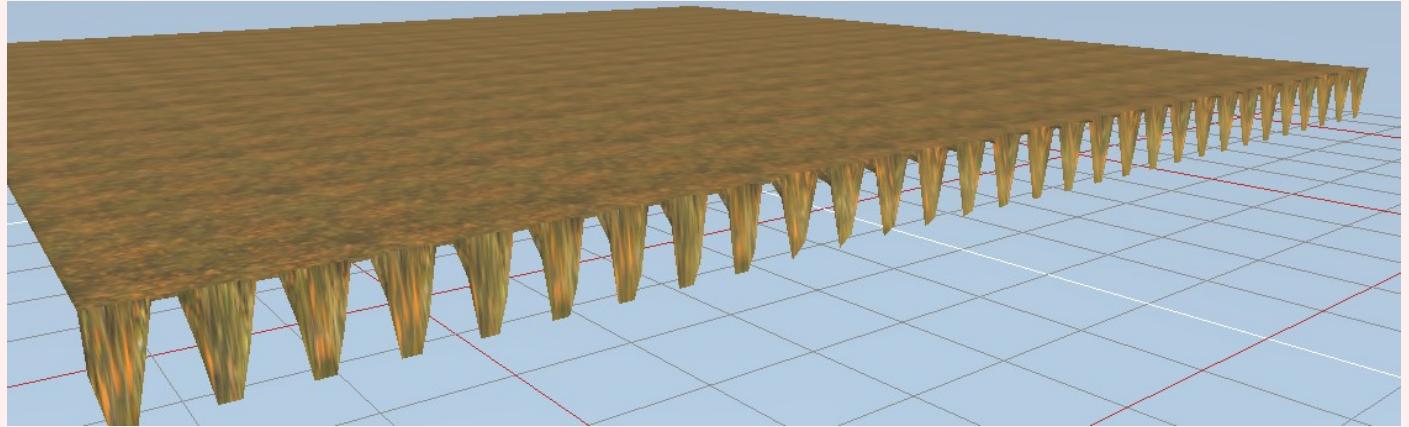
The **Falloff** parameter still has the same purpose; it sets how much of the brush is dedicated to the falloff region.

If you just want everything to be perfectly flat, it is fine to set the **Value** to 1.00 and the **Falloff** to 0.00. However, if you want a more subtle effect (e.g. at the edges of the flattened region), I recommend a reduced **Value** and a higher **Falloff**. For example, a **Value** of 0.50 and a **Falloff** of 0.80 allows me to stroke overlapping loops with my brush that result in perfectly flat terrain where I've been looping and smooth slopes at the edges.

In the **Flatten** mode, the brush preview disc is drawn in red for a **Value** above 0.50, black for a **Value** below 0.50.

Bug: Normally the brush preview disc increases in intensity as the **Value** changes away from some neutral value that has no effect when painted. In **Flatten** mode, however, the brush preview disc treats 0.5 as the neutral **Value** when 0.0 is the actual neutral **Value**. This means that the preview disc is invisible near a **Value** of 0.50 even though this **Value** causes a large effect.

Bug: A freshly created map is mostly at a single middling height, but one edge often has some weird height drops. Use `Flatten` mode to fix it.



Smooth Brush Mode

In the `Smooth` mode, the target height at each point under the brush is a weighted average of the heights around the edge of the brush. E.g. the target height in the middle of the brush is exactly the average of the heights around the edge of the brush, but the target height near the left side of the brush is closer to the height of the left edge than the right edge. In other words, `Smooth` mode tries to create a smooth slope.

The `Size` parameter still has the same purpose; it sets the radius of the paint brush.

The `Value` parameter indicates the magnitude of the effect. In this case, a `Value` of 1.00 does not fully smooth the terrain as soon as you click, but it takes very little dragging of the brush to achieve a completely smooth slope.

The `Falloff` parameter again specifies the falloff characteristics of the brush. In this mode, however, falloff is counter-productive. The results can be very non-smooth if terrain is fully smoothed at the center of the brush while the terrain at the edges stays close to its original heights.

Because of the way the Smooth brush works, it is perfectly reasonable to set the `Value` to 1.00 and the `Falloff` to 0.00.

Bug: Normally the brush preview disc increases in intensity as the `Value` changes away from some neutral value that has no effect when painted. In `Smooth` mode, however, the brush preview disc treats 0.5 as the neutral `Value` when 0.0 is the actual neutral `Value`. This means that the preview disc is invisible near a `Value` of 0.50 even though this `Value` causes a large effect.

Brush Keyboard Shortcuts

Keyboard shortcuts are listed on the **Brush** dialog box. E.g. **shift+E** decreases the **Size** by a step, and **shift+R** increases it by a step. Each step is 5% of the slider length, but remember that the slider has an exponential relationship to the **Size** value.

Shift+V is for the Randomize checkbox, which is disabled for all **(Geometry)** brush modes.

Ctrl+V rotates among the **(Geometry)** brush modes: **Height**, **Flatten**, and **Smooth**.

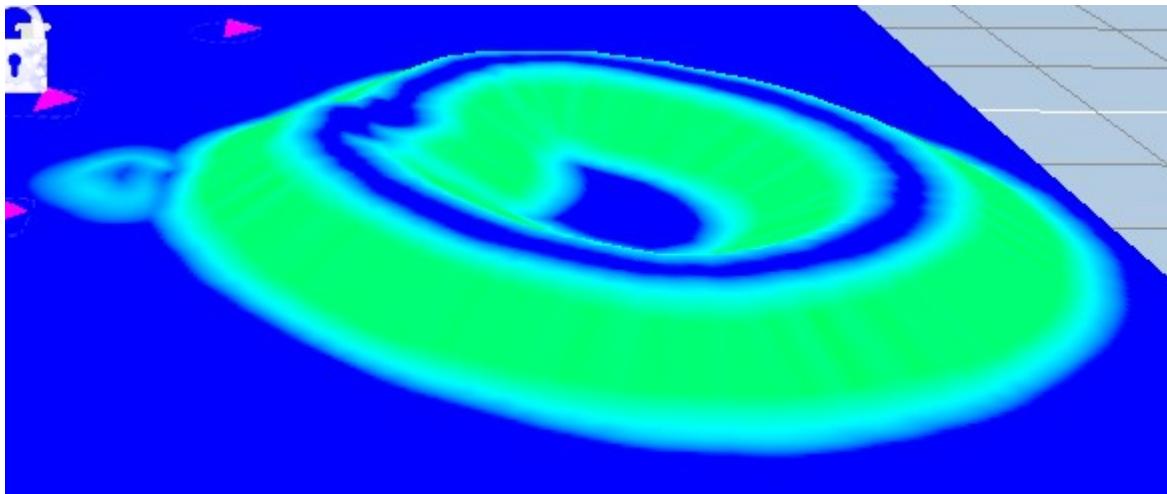
Ctrl+Z is an unlisted shortcut that corresponds to the **Undo Current Changes** button.

The keyboard shortcuts can be used not just when the **Brush** dialog box has focus but also when the main panel or the **Scene View** have focus. In other words, they can be used essentially any time you are using a brush. If you learn and use these shortcuts, your painting speed will be much faster with all brushes.

Height Information

Whenever the mouse is over the terrain in the main panel, the editor annotates the current terrain height and angle in the upper left corner. The angle is 0° for flat terrain, increasing to near 90° for a cliff.

You can get a quick read of the terrain angles at all points at once by enabling **Show HeightMap Angles** in the toolbar.



The heightmap colors correspond as follows:

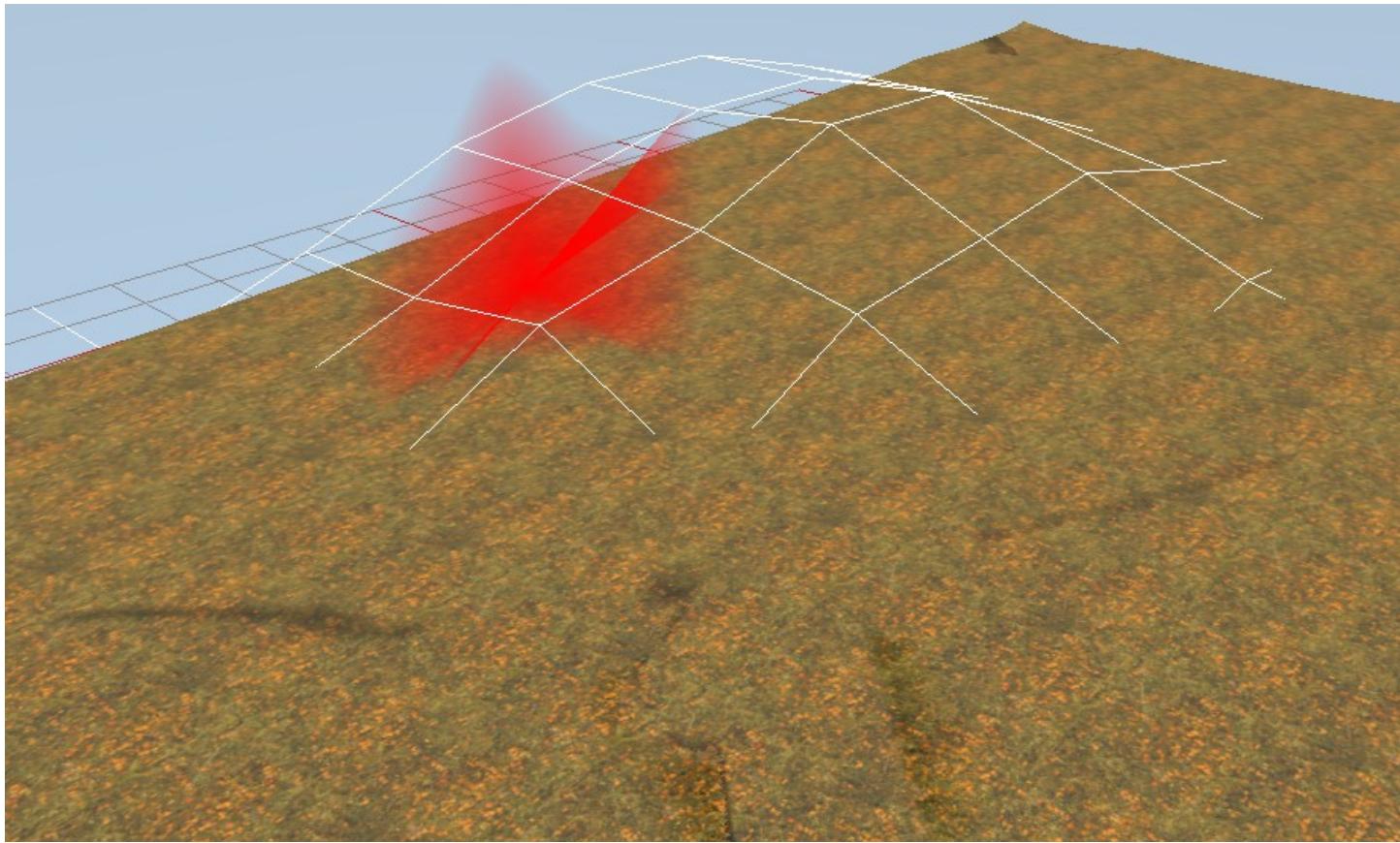
- dark blue: 0°
- cyan: 1° – 24°
- green: 25° – 34°
- yellow: 35° – 44°

- orange: $45^\circ - 59^\circ$
- red: $60^\circ - 90^\circ$

In ideal conditions in SnowRunner, a well-prepared stock can climb a slope up to about 60° . For comparison, according to the Guinness World Records, the steepest street in the world is 19° .

Deceptive Height Cues

While editing the terrain height, the Editor changes the way it draws the terrain. However, it doesn't change the way it draws everything else until you commit your changes and rebuild the terrain. For example, I have (almost) completely flattened the terrain in the screenshot below, but it doesn't look flat at all.



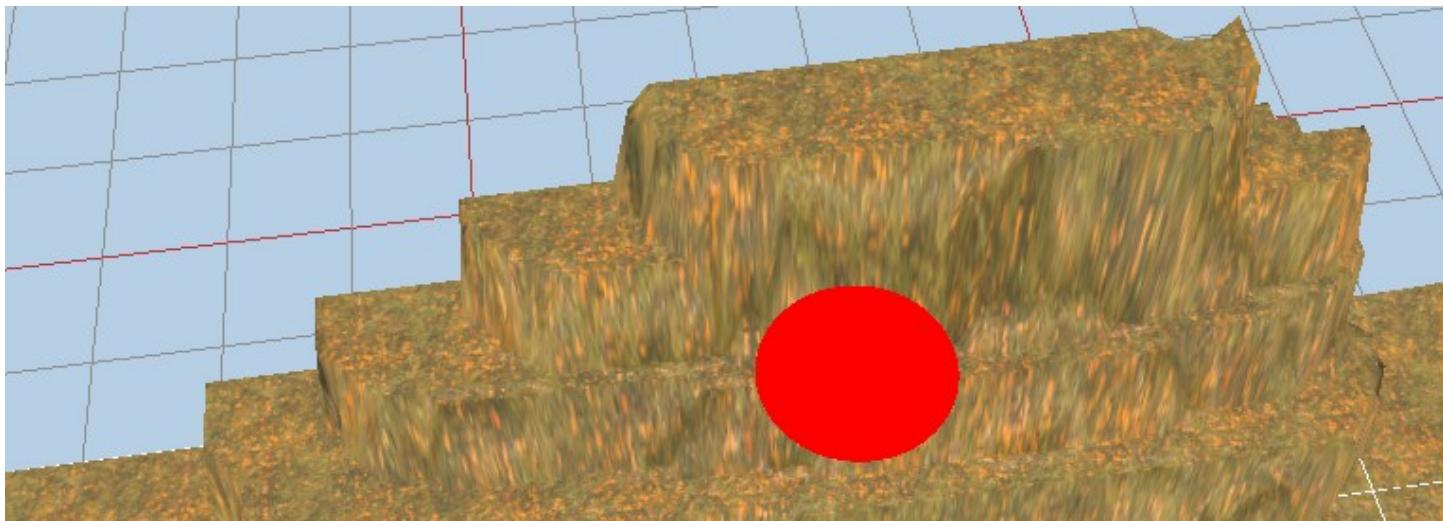
The deceptive cues above are as follows:

- The terrain is still colored with its original shadows, showing where steep slopes used to exist.
- The base of the terrain blocks (the white wireframe mesh) remains at its original height.
- The brush preview disc is warped to follow the contours of the original terrain.

Another deceptive cue not shown here is the height and angle displayed in the upper left of the main panel, which continue to show the values for the original terrain.

The only cue that isn't deceptive is the edges of the map. The map edges are straight, which implies that the terrain must be flat (or at least smooth). A couple of hitches near the corner imply that I didn't quite get it flat there.

The brush preview disc is particularly deceptive because its horizontal position is determined by where I'm pointing on the original terrain, which can be very different from the horizontal position where I'm pointing in the edited terrain. In the screenshot below, I've built up the terrain from a flat plane. The brush preview disc appears to be pointed at the middle of the slope, but it actually represents a position right at the edge of the map. (Compare the disc position to the virtual line connecting the flat edges of the terrain at the left and right.) Drawing at this position will actually draw at the very edge of the map at the top of the slope.



Rotating the camera and watching the motion of the terrain can help a little. But really the best solution is to commit your big changes, see how they look, and then make smaller changes to fix up the details.

Navigate While Painting

While right-click-dragging, it can be frustrating to run into the edge of the main panel. If the mouse leaves the main panel and re-enters somewhere else, the brush paints a straight stroke from where it left to where it re-entered.

To avoid this, you can navigate while a brush stroke is in progress. While continuing to hold down the right mouse button, you can then press the control key and the left mouse button to switch to dragging the terrain around. If you then release the left mouse button while continuing to hold the right button down, you can resume your paint stroke at the same position on the terrain, but a new position within the main panel.

This trick only works well with control + left-drag because the mouse moves exactly with the terrain. With a regular left-drag or with shift + left-drag, the terrain moves away from the mouse, so you end up with a stray brush stroke when you release the mouse.

Restore a Previous Terrain State

Click `Undo Current Changes` in the `Brush` dialog to revert your changes without updating the underlying bitmap file. Changes are reverted across all brush modes back to the point when you first entered the `(Geometry)` brush. However, the Editor doesn't keep any further backups of previous files.

I recommend that if you're happy with your terrain, manually make a copy of the `prebuild` directory to an archival location. Then, if you accidentally mess something up, you can simply restore one or all of the archived terrain bitmap files back into your `prebuild` directory.

Unlike when you update the map's XML file, the MudRunner Editor won't automatically notice changed bitmap files. Right click in the main panel and select "Rebuild Terrain" to redraw the map from the files.

Mud

Mud is tricky to get right. Expect to experiment a lot to get a feel for it.

Note that many of the terrain materials are already a little soft, including the default dirt material. This means that if a truck drives over it repeatedly, it will eventually cut ruts into the terrain and lose traction. Adding mud makes the material even softer and more slippery.

Mud can be painted in four different ways:

- The `(Geometry) → (Mud)` brush in `Extrudes` mode paints mud that is hard to see.
- The `(Geometry) → (Mud)` brush in `Automatic` mode can paint wide swaths of visible mud.
- The `(Geometry) → (Mud)` brush in `Automatic` mode can also paint narrow muddy ruts that look like a truck drove through.
- The `(Geometry) → (QuickMud)` brush paints a wide area of mud that appears churned by many trucks.

Extrudes Mode

Expand the `(Geometry)` category in the `Scene View` and click `(Mud)` to start using the mud brush. The default mode is `Automatic`, but we'll start with the `Extrudes` mode.

The mud brush allows painting with relatively fine detail. For a particular `Size` value, a mud brush is half the size of any other terrain brush. In other words, the mud brush size effectively measures the brush diameter, whereas all other brushes measure the brush radius.

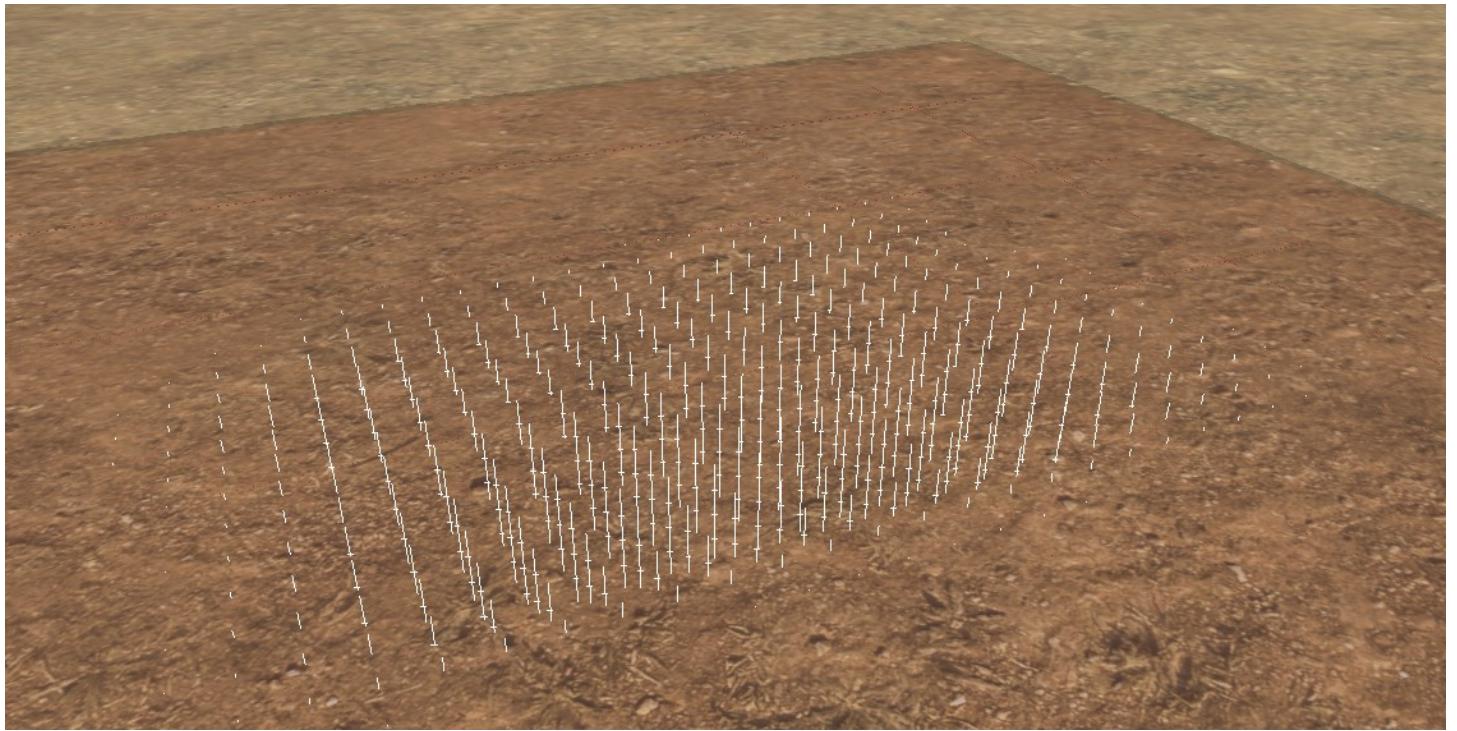
A `Value` greater than 0.50 increases the amount of mud, and a `Value` less than 0.50 decreases the amount of mud. When increasing mud in `Extrudes` mode, the brush preview disc is red. When decreasing mud, the brush preview disc is black.

Bug: Using 0.50 as the neutral `Value` is a holdover from SpinTires and MudRunner where all brushes used the same slider range. It would make more sense to have a range of -1.00 to +1.00 with a neutral value of 0.00. This neutral value of 0.50 is common to many of the brushes, so you'll eventually get used to it.

Adding mud with the `Extrudes` brush softens the terrain in the painted area, making it easier for a truck to sink into it. By default, mud painted with the `Extrudes` brush is completely invisible. However, while the mud brush is active, the Editor indicates the presence of mud with two visual cues:

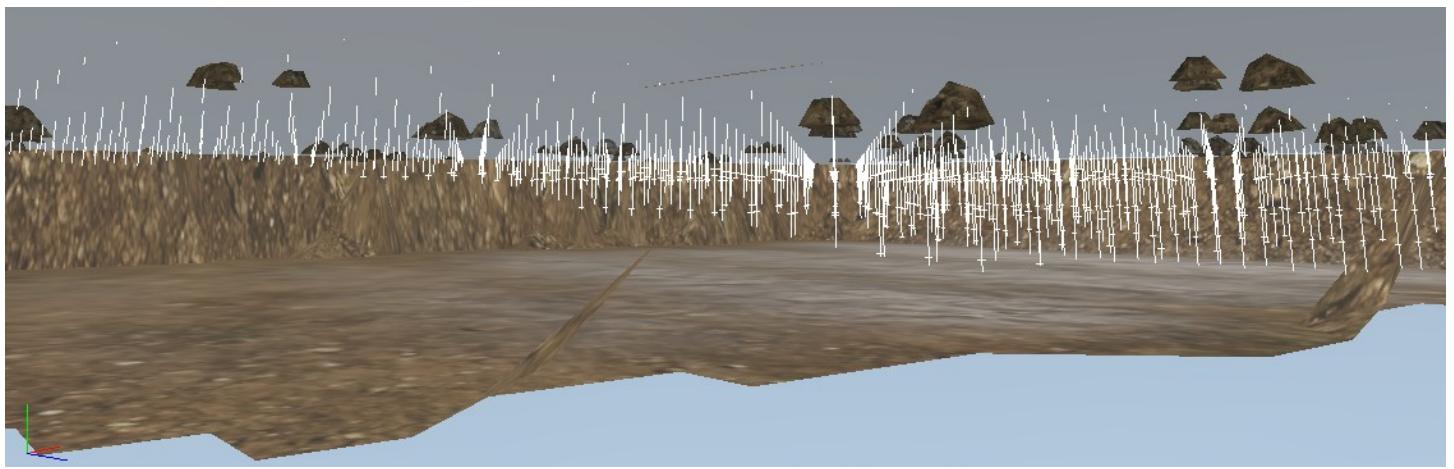
- A brown tint covers a large block of terrain, indicating that mud is present.

- Thin white lines extend down from the terrain surface where mud was painted. The length of each line represents the softness of the mud in that area. A tick mark is placed on the line for every 25% of the maximum mud softness that is present. E.g. if a line has one tick mark near the bottom, the mud is just over 25% of the maximum possible softness. This is generally passable even by 2WD vehicles with highway tires. Two tick marks is about 50% of the maximum softness, and is generally passable by AWD vehicles with offroad tires. Beyond that, only the toughest, most mud-capable trucks will find passage.



If you left click to exit the mud tool, there is (by default) no visual indication of the mud at all. This is true within the game as well. I'll address the visibility of **Extrudes** mud in the next section.

While a mud brush is activated, you can move the Editor camera slightly underground to see a bit more about how mud is handled in the game. Wherever mud is present, a second terrain layer is added beneath the main surface. This second layer provides another surface for both visual rendering and for physics when the truck breaks through the main surface and starts churning up the mud.



Bug: The first time you paint a new section of mud, the main surface is temporarily erased and only the second, underground layer is rendered. If you exit the mud brush and re-select it, the main surface is drawn correctly.

This second terrain layer highlights an important point: although the thin white lines imply a mud depth, they really only indicate its softness. If a heavy truck persists in churning up the mud, it can always dig all the way to the second layer.



In fact, the game automatically creates a second layer wherever the truck starts churning up dirt. With enough persistence, it is possible to dig deep ruts even without mud.

Tip: Avoid sudden transitions between deep mud and no mud. The truck will bounce as if it hit a low wall as it is suddenly forced up and over the solid dirt.

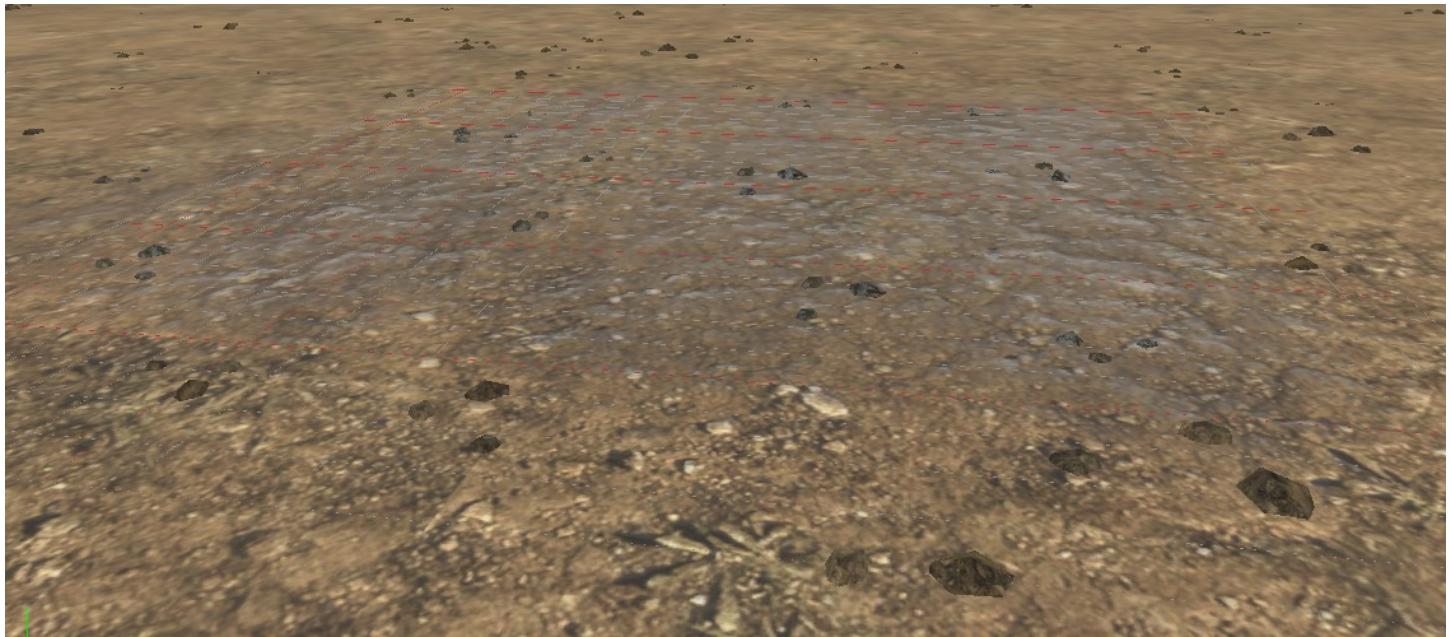
Extrudes to Wetness

The `(Terrain)` feature in the `Scene View` has a property called `Extrudes to Wetness` with a default value of 0.0. This property causes wetness to be added to the terrain in proportion to the `Extrudes` softness. Wetness provides a subtle visual indication that mud is present. It also makes the mud more slippery.

The `Extrudes to Wetness` value is multiplied by the mud softness (as a fraction of the maximum mud softness) to get the derived wetness (as a fraction of the maximum wetness). The resulting calculated wetness is capped at 100% if necessary.

Tip: Saber recommends an `Extrudes to Wetness` value of 1.0.

Newmud painted in `Extrudes` mode is rendered with the current `Extrudes to Wetness` value. But to apply the parameter to existing mud you must select `Rebuild Terrain` from the context menu.



A second property just below `Extrudes to Wetness` sets the `-threshold` at which it applies. Mud that is less deep than the `-threshold` value (as a fraction of the maximum mud softness) has no wetness applied. The wetness is unaffected for mud that is deeper than the threshold.

Tip: Saber recommends a `-threshold` value of 0.20 for a winter environment and a smaller value such as 0.05 for a warmer environment.

Automatic Mode

The `(Mud)` brush also has an `Automatic` mode that is used to draw obvious mud. Its behavior is different if the `Size` is greater than 2.5 meters vs. less than or equal to 2.5 meters.

As in the `Extrudes` mode, a `Value` greater than 0.50 increases the amount of mud, and a `Value` less than 0.50 decreases the amount of mud. When increasing mud in `Automatic` mode, the brush preview disc is white. I'll describe decreasing (erasing) `Automatic` mud in a later section.

The `Falloff` parameter is disabled in `Automatic` mode because paint strokes have their own unique behavior in this mode.

Automatic Mode With Size > 2.5

`Automatic` mode is called “automatic” because the brush does multiple things at once. With `Size` > 2.5 it does the following.

- It increases the softness of the mud in the same way as the `Extrudes` brush.
- It adds a dark tint to the painted area to indicate where subsurface soil has been exposed.



Since the `Automatic` mode records mud softness in the same way as the `Extrudes` mode, the `Extrudes to Wetness` property applies to both. So for mud that is deep enough, it can have both the dark tint of subsurface soil, plus additional tint and shininess from wetness.

Automatic Mode With Size ≤ 2.5

With size less than or equal to 2.5 meters, the Editor adjusts **Automatic** mode to paint a rut as if a truck drove through. This increases the softness of the mud and adds a dark tint as above, and it also adds lumpiness to the surface where the mud is pushed down in the center of the rut and pushed up slightly on each side of the rut.

Tip: Saber recommends a **Size** of 0.5 – 0.8 meters to represent a rut from a truck with narrow or wide tires.

Tip: Saber recommends a **Value** of 0.54 – 0.7 to represent the range from very shallow to very deep ruts.

Tip: A pair of parallel ruts best simulates the passage of a single truck (or of multiple trucks driving the same path). Position a truck of the appropriate size near the ruts to gauge how far apart they should be.

Tip: The **Autofade** checkbox is well suited for painting a rut that begins and ends with a smooth transition. However, it takes a steady hand to get the **Autofade** tails to be parallel for each pair of ruts.



With the recommended values, the mud in the ruts is not particularly soft, and it won't generate much wetness from the **Extrudes to Wetness** parameter. A truck driving through these ruts is more effected by the lumpiness of the ruts than the softness of the mud. You can then use the **Extrudes** mode to increase the softness of the mud and/or manually add wetness to make it more slippery.

Decrease Automatic Mud

Mud can be decreased or erased with a brush **Value** less than 0.5. The effect is different depending on whether the brush is in **Extrudes** or **Automatic** mode.

The **Extrudes** mode only directly effects the mud softness. The lumpiness and dark tint are unaffected until the mud softness reaches zero, at which point the lumpiness and tint are removed. However, the lumpiness and tint remain recorded and will return if the **Extrudes** mode is used to soften the mud again.

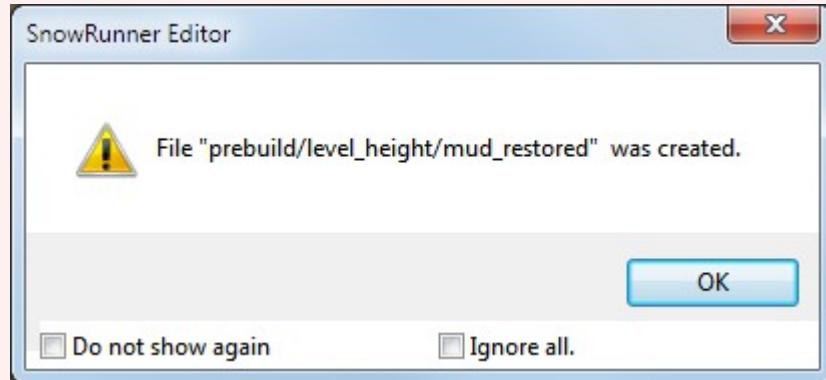
The **Automatic** mode reduces the mud softness, its lumpiness, and the dark tint simultaneously. However, it does a really bad job reducing the lumpiness and tint. Depending on how you apply the brush, you can end up removing the tint while leaving the lumpiness, or remove the deep pat of the rut while leaving the pushed up parts, etc. If you aren't satisfied with the visual appearance of a rut, it is generally easier to entirely erase it and start again. Use **Automatic** mode with **Value** 0.0 and a **Size** large enough to entirely cover the affected area. This also erases the mud softness, so you'll need to reapply that with the **Extrudes** mode if desired.

Note that the lumpiness from the mud is recorded separately from the base terrain height. Erasing the mud restores the original terrain appearance.

Restore a Previous Mud State

As in other terrain brushes, you can click **Undo Current Changes** in the **Brush** dialog to revert your changes without updating the underlying bitmap file. Changes are reverted across all brush modes back to the point when you first entered the **(Mud)** brush.

Bug: All **(Geometry)** items in the **Scene View** have a context menu item called **Restore version "mud" from "data.stg"**. Since **data.stg** is only updated on a save, this appears perfect, but unfortunately it doesn't work. Selecting the action pops up a dialog to indicate that it is done:



If I replace my **mud** file with the new **mud_restored** file and reload the map, it appears to have worked (although with some leftover wetness tints that are presumably stored separately). However, rebuilding the terrain reverts back to the unwanted mud state. I suspect that the “restore” process is actually using the latest unsaved **data.stg** in memory rather than the file on disk.

I recommend that if you're happy with your mud, manually make a copy of the **prebuild** directory to an archival location. Then, if you accidentally mess something up, you can simply restore the archived **mud** file back into your **prebuild** directory.

QuickMud

Expand the [\(Geometry\)](#) category in the [Scene View](#) and click [\(QuickMud\)](#) to quickly paint a large swath of heavily churned mud. This feature works differently than other terrain brushes because you don't have direct control of the results. Instead, you simply indicate the areas where you want mud, and the Editor decides how it should look.

A [Value](#) greater than 0.50 increases the amount of mud, and a [Value](#) less than 0.50 decreases the amount of mud. When increasing mud, the brush preview disc is red. When decreasing mud, the brush preview disc is black.

The figures below show the act of painting with the [QuickMud](#) brush and the result.



Essentially, you paint an area with a red tint, and the Editor creates many pairs of truck ruts at random orientations in order to churn that area (and somewhat beyond, since it uses autofade).

The mud generated by the [QuickMud](#) brush is kept separate from the mud created by the [Mud](#) brush, so you can't directly observe the softness of the generated mud. However, experimentation reveals that where the [QuickMud](#) tint is brightest, the mud randomly ranges in softness from 0% to 35% of the maximum softness. Where the [QuickMud](#) tint is weaker, the maximum softness is reduced.

The brightness of the [QuickMud](#) tint also appears to have a subtle effect on the visual depth of the generated ruts, but I'm not completely positive about that. At the very weakest tints, [QuickMud](#) generates light patches of mud tint without any ruts.

Materials

The **PbrMaterials** category in the **Scene View** holds one or more materials that make up the surface of the terrain. These materials include surfaces such as grass, dirt, gravel, sand, asphalt, and snow. Each possible surface is referred to as a texture layer.

A texture layer obviously describes the texture of the terrain, i.e. the bitmap that is applied to the terrain. It also enables associated 3-D objects such as grass and pebbles to set on the surface of the terrain.

The texture layer also describes intangible aspects that can't be seen in the Editor, e.g. whether dust can be kicked up from the terrain and what it sounds like when tires skid on the terrain.

*** For ease of description, I'll mostly refer to the materials' impact on the texture of the terrain. The impact on other aspects is implied.

One terrain block (24m×24m square of the terrain) can use only one material, but that material may include up to four texture layers arranged however you like across that terrain block. I'll begin by describing how texture layers are handled in one material before we start adding more materials.

Material Properties

Your map is initialized with its first material, called [\(PBR Terrain Material\)](#) under [PbrMaterials](#) in the [Scene View](#). Click on it to bring up its properties. This also brings up a brush tool, but we'll deal with that in the next section.

Name	
<input type="checkbox"/> Material	
Albedo wetness mult	1.000000
Roughness wetness mult	1.000000
Mask channel index	0
<input type="checkbox"/> Layer 1 (base)	
File	default
[Choose File]	[press]
Tiling scale	1.000000
<input type="checkbox"/> Layer 2	
File	
[Choose File]	[press]
Tiling scale	1.000000
HM blending contrast	0.900000
Mask channel index	-1
<input type="checkbox"/> Layer 3	
File	
[Choose File]	[press]
Tiling scale	1.000000
HM blending contrast	0.900000
Mask channel index	-1
<input type="checkbox"/> Layer 4	
File	
[Choose File]	[press]
Tiling scale	1.000000
HM blending contrast	0.900000
Mask channel index	-1

The [Name](#) property is blank, which is why this material has the default name of [\(PBR Terrain Material\)](#). You can enter any name you want for the [Name](#) property. This [Name](#) is displayed in the [Scene View](#) to help you keep track of the purpose of this material.

The [Albedo wetness mult](#) property influences how much that wetness darkens the tint of the terrain. A higher value leads to darker wetness. A value of zero removes the tinting effect of wetness.

The [Roughness wetness mult](#) property influences how wetness is displayed on the terrain. A higher value makes wet terrain look smoother. A value of zero removes the smoothing effect of wetness.

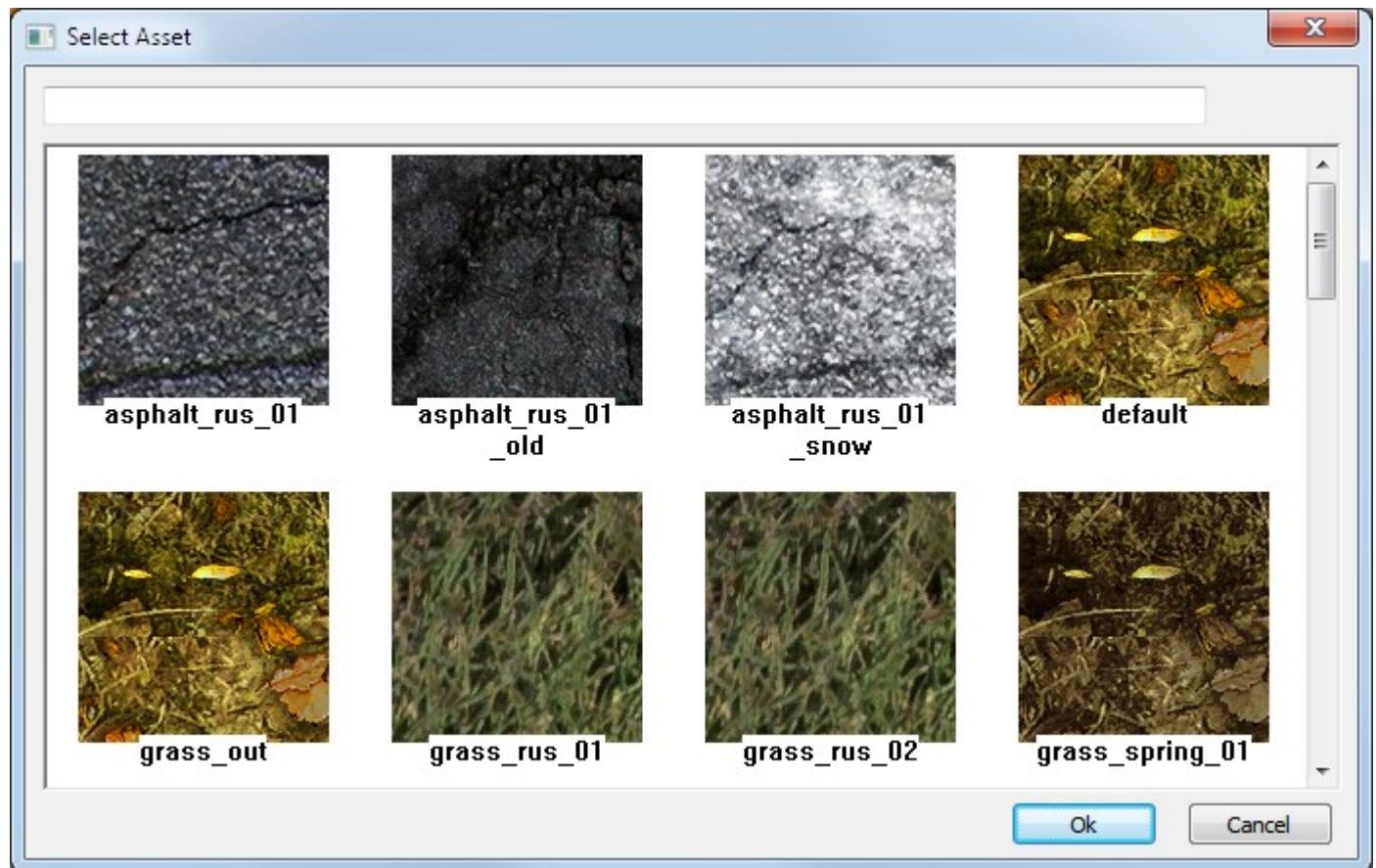
Tip: Saber recommends for winter environments `AlbedoWetnessMult` = 1.0 and `RoughnessWetnessMult` = 1.0. They recommend for warmer environments `AlbedoWetnessMult` = 1.0 and `RoughnessWetnessMult` = 1.0.

The `Mask channel index` cannot be edited and is always 0 for the first material.

Texture Layer Properties

The remaining properties for the material are specific to each texture layer.

Choose the texture for a layer by clicking `[press]` next to `[Choose File]`. Choose your desired texture from the dialog box that pops up.



By default, four copies of the texture are tiled in a 2×2 pattern to fill each terrain block. But for textures that require finer detail, we want smaller tiles, with more of them to fill a block. The `Tiling scale` property specifies how much smaller each copy of the texture should be. E.g. a value of 5 shrinks the texture by a factor of 5, so it takes a 10×10 pattern of tiles to fill a terrain block. The value does not need to be an integer. If the edge of a terrain block is reached in the middle of a tile, the tile continues smoothly into the next terrain block as long as it uses the same texture.

Bug: The proper `Tiling scale` is always the same for each particular texture, but instead of specifying the property with the texture itself, the Editor makes us specify the property every time we use a texture. If we forget to do that, the result can look very weird. E.g. enormous leaves in the default texture.

Tip: A good-looking `Tiling scale` is **not** 1.0 for most terrain layers. See the Texture Layer Reference section on page 162 for recommended values.

Layers 2 – 4 have additional properties:

When different textures within the same material are placed next to each other on the terrain, the edges of those textures are blended together to avoid a sharp demarcation. The `HM blending contrast` property specifies the sharpness of that blend line. The effects of this property are described in more detail in the Edge Blending section on page 91.

Bug: The proper `HM blending contrast` should be specified where the texture is described, not where it is used.

Tip: Saber recommends the following `HM blending contrast` values for various textures:

- soft surfaces: 0.7 – 0.8
- hard surfaces: 0.8 – 0.9

The `Mask channel index` cannot be edited and is always 2 less than the layer number. E.g. layer 3 uses `Mask channel index` 1. This per-layer property is separate from the per-material `Mask channel index` described above.

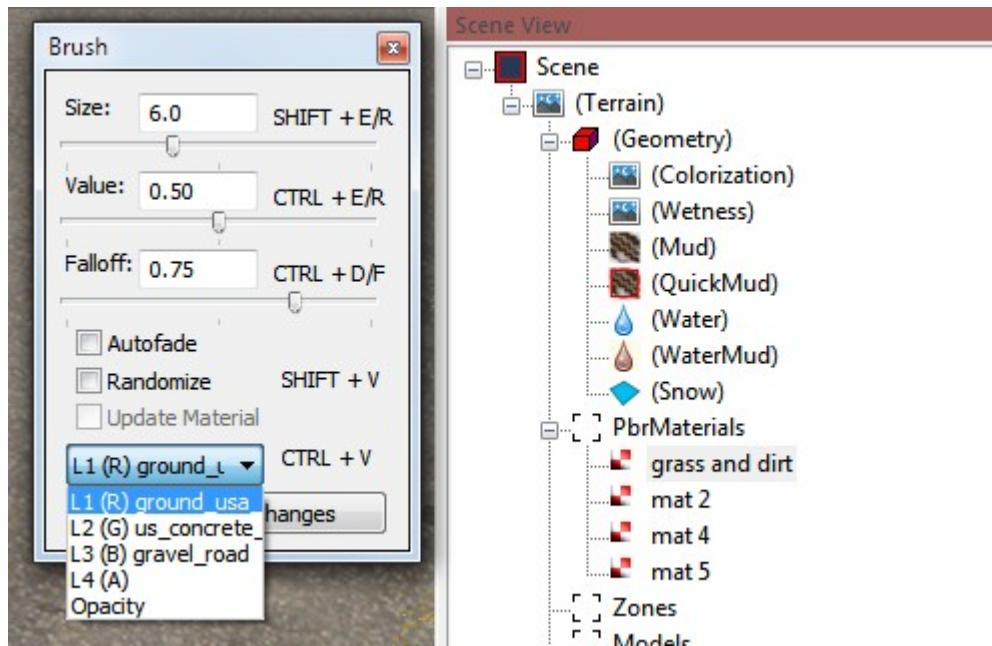
Paint a Texture Layer

Texture is painted onto the terrain in 0.5×0.5 meter squares. Each square typically displays the characteristics of a single texture layer, but some blending can occur as described in the sections below.



Textures are painted onto the terrain in layers like separate coats of paint. Each texture uses a stencil so that it is painted in some squares but in others. Whichever texture is painted last is most visible, but a weakly painted texture will allow color to bleed through from the underneath textures.

To begin painting, click the desired material under **PbrMaterials**, then select the desired texture in the mode selection dropdown of the **Brush** dialog box.



The bottommost texture is always painted across the entire terrain block, so it cannot be selected in the **Brush** dialog. Each of the three other textures in the material can be selected and painted over top.

Bug: The three other textures are labeled **Layer 2**, **Layer 3**, and **Layer 4** in the property panel, but they are confusingly abbreviated **L1**, **L2**, and **L3** in the Brush dialog.

The **L4 (A)** brush mode is used only for snow. The **Opacity** brush mode is used to select which material to apply. I'll describe each of these in later sections.

A **Value** greater than 0.50 increases the strength of the texture, and a **Value** less than 0.50 decreases the strength of the texture. When increasing the strength, the brush preview disc is red (**R**), green (**G**), or blue (**B**), depending on which texture is selected. When decreasing the strength, the brush preview disc is black.

Remember that the paint is layered, so if you paint a strong **Layer 4** (**L3**), nothing you paint in the other layers will be immediately visible. However, if you then erase some of the paint from **Layer 4**, your changes to the lower layers become visible.

Value Blending

Where a texture layer is not painted at full strength, the display engine blends it with the lower texture layers. There is some blending at the pixel level, but mostly the engine looks for the implied high spots in each texture and gives them priority for display

The below screenshot shows an extreme close-up of a gravel texture that is painted only weakly over a grass and leaves texture in the center left and a concrete texture in the center right. The images of the gravel pebbles are dominate the grass and leaves, but the grass and leaves texture is stronger in the gaps between gravel pebbles. Interestingly, even though the gravel is painted over the concrete, our eyes interpret it as a thin layer of concrete partially covering the low spots in the gravel.



Edge Blending

When a layer of paint has little falloff between regions of “paint” and “no paint”, then blending with the lower layers is based on the `HM blending contrast` property of the upper layer. The below screenshots show the same edge in which the concreate has an `HM blending contrast` of 0.6 on the left and 1.0 on the right. With a value of 1.0, the edge wiggles a bit to follow the implied high spots in each texture, but colors are never blended between the two textures.



These screenshots also illustrate that even though paint is applied in 0.5×0.5 meter squares, the display engine rounds off corners to avoid an unnatural-looking sharp bend. It also uses diagonal lines to connect paint squares that are painted in a stairstep fashion.

When the `HM blending contrast` property value is below about 0.55, lower texture layers start to bleed through even where the upper layer is at full strength. See the Material Properties section on page 86 for recommended values.

Bleed of 3-D Objects

A texture layer can have associated 3-D objects such as grass and pebbles. After painting or removing a texture layer, select `Rebuild Terrain` from the context menu to redraw these 3-D objects in the correct places.

Where two texture layers are smoothly blended, 3-D objects from each texture can be intermixed depending on the strength of each paint.

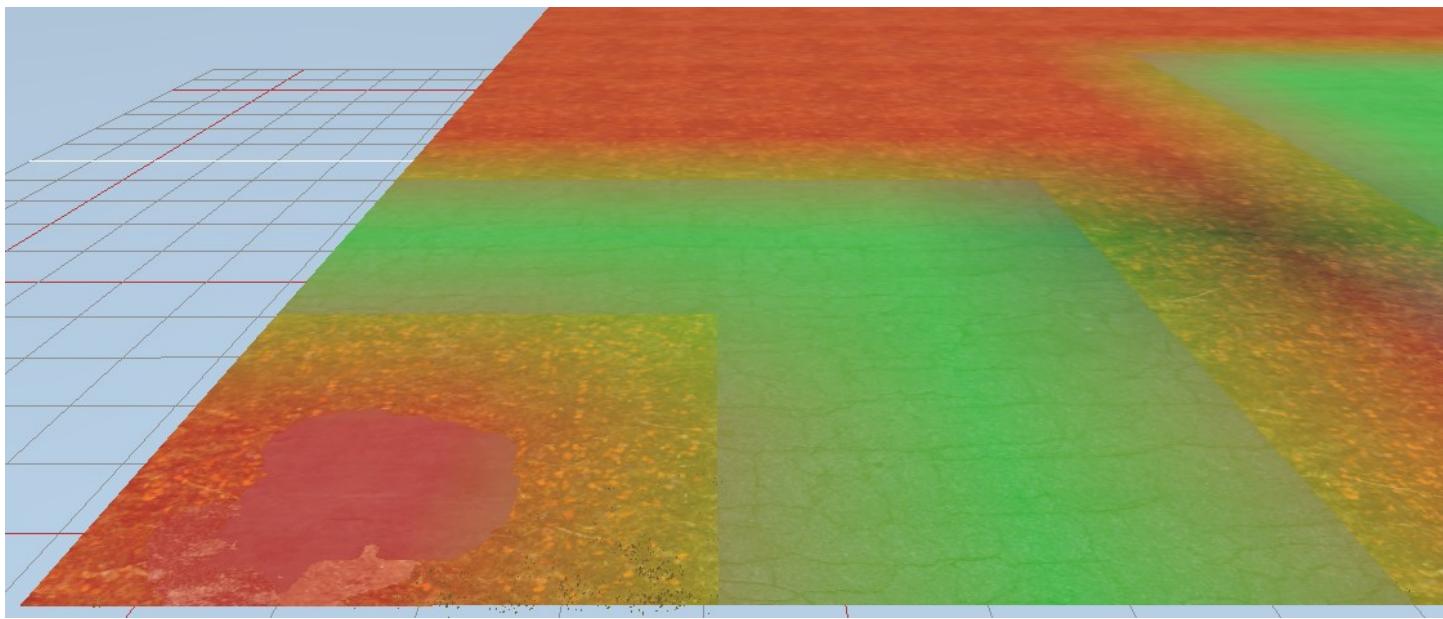
Where two texture layers have a sharp line dividing them, a 3-D object from one may intrude on the other simply because of the size of the 3-D object. This is especially noticeable for grass when viewed closely. However, it is much less noticeable with the typically high camera position while driving in the game.



Additional Materials

To add additional materials, select `Add Pbr Material` from the context menu. Each material can have its own combination of texture layers.

Each terrain block can use only a single material. The material used for each terrain block is chosen by painting the material onto that terrain block. To begin painting, click the desired material under [PbrMaterials](#), then select [Opacity](#) in the mode selection dropdown of the [Brush](#) dialog box. Because the default texture layer may be shared across different materials, the Editor helps distinguish which material is used where by tinting them in different colors.



Each map can use up to four different materials. While painting with [Opacity](#), each material is tinted with a different color:

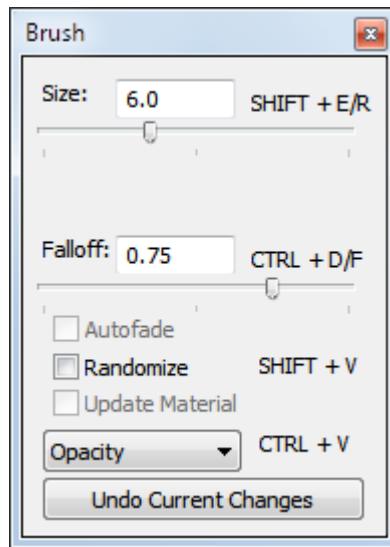
- The first material is tinted red.
- The second material is tinted green.
- The third material is tinted blue.
- The fourth material is tinted black.

Bug: Although each material has a distinct boundary at the edge of a terrain block, the tinting is smoothly blended between the centers of terrain blocks. This can make it harder to discern the true tint of small regions of material.

Although you can add more than four materials and attempt to paint with these additional materials, the changes will revert as soon as you [Rebuild Terrain](#).

Bug: Although Saber documents that the Editor supports up to four materials on a map, painting with the fourth material doesn't work correctly. If you [Rebuild Terrain](#), these terrain blocks revert to the first material instead. This can be fixed up by hand with some tricky bitmap editing outside of the Editor. See the Material Layout Bitmap section on page 174 for details.

Painting with **Opacity** is all or nothing, so there is no **Value** parameter. The **Size** allows you to potentially paint many terrain blocks at once. Oddly, the **Falloff** parameter can still be adjusted, but it does nothing.



To paint the selected material onto the terrain with **Opacity**, right click on the desired block. Or right click and drag to quickly paint multiple blocks. There is no brush preview disc. As long as the **Size** is small, it's simply the terrain block under the mouse. If the **Size** is large, it is the many terrain blocks within that range of the mouse.

Material **Opacity** is somewhat different than other painting modes in that selecting a different material requires a new selection from the **Scene View** rather than a quick change to the **Value** or brush mode.

Bug: While painting with **Opacity**, right click updates both the artificial tinting and the textures on the terrain to match the new material. However, **Undo Current Changes** only reverts the changes to the tint. To correctly revert the textures on the terrain you must then select **Rebuild Terrain** from the context menu.

When you change the material assigned to a terrain block, the paint stencils for layers 1 – 4 are retained, although they may now point to different textures in the new material.

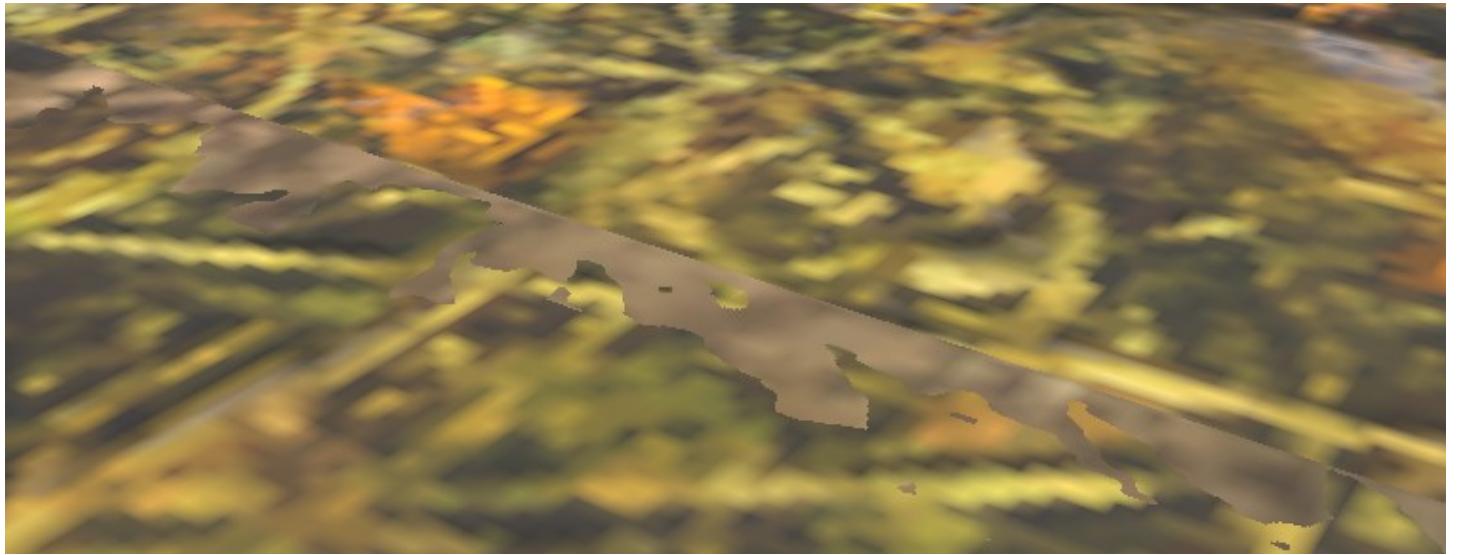
Texture Edges Between Materials

Different terrain blocks can use different materials and thus different combinations of texture layers. However, adjacent terrain blocks must have at least one matching texture which is used along the boundary between the terrain blocks. Otherwise, a sharp line results where the texture suddenly switches from one block to the other.



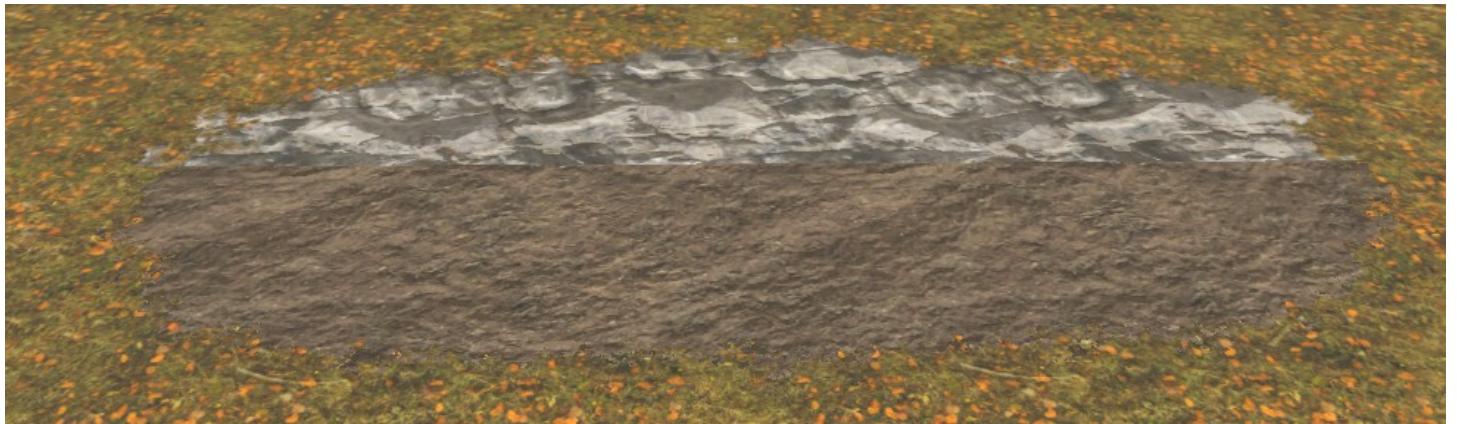
Textures can also be blended among layers at boundary between terrain blocks, but only if the materials in both blocks use the same textures in all blended layers. Needing to share multiple textures across materials sharply limits how many different textures can be used throughout the map, so typically you will share only one texture and avoid any blending of textures near material boundaries.

BTW, if two adjacent materials use the same texture, but in different layers, the display engine will blend between the layers separately on each side of the boundary. I.e. if a block uses grass in layer 0 adjacent to grass in layer 1 in an adjacent block, the first block displays a blend of layer 0 and layer 1 near the boundary. Since layer 1 is some other texture, this results in color bleed from an unwanted texture and a sharp line at the block boundary. Therefore, textures shared across materials should always be shared in the same layer.



If four materials are used in a map, and each material holds up to four texture layers, but at least one texture is shared across each pair of materials, then the maximum total number of textures per map is $4+3+3+3 = 13$.

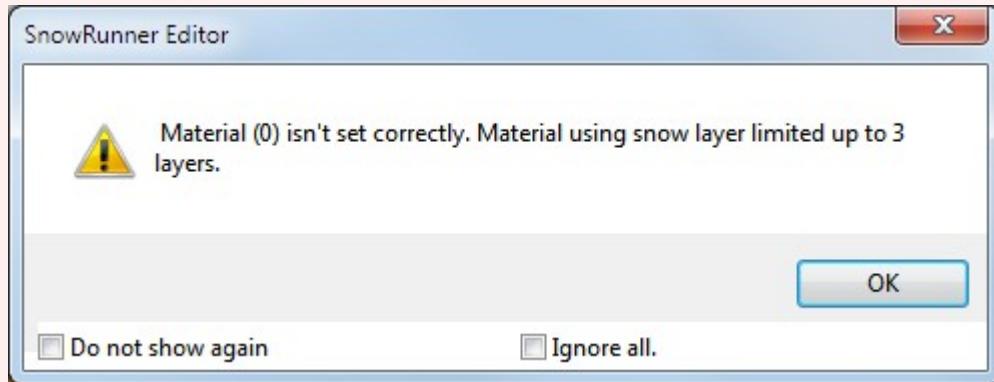
Caution: Although you select a particular material to begin painting a texture layer, the brush actually paints all materials with the selected texture layer number. So if you paint across the boundary of two materials, the painted texture takes on the characteristics of whatever texture is assigned to that layer in each material.



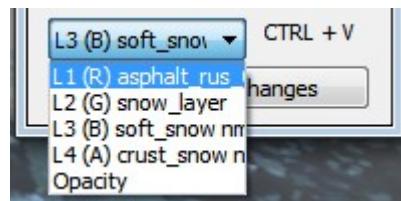
Snow Layers

The `snow_layer` texture is special and is officially supported only in `Layer 3` of a material. When `snow_layer` is used in `Layer 3`, `Layer 4` must not have any texture assigned to it.

Bug: There is no way in the Editor to remove a texture that has been assigned to a layer, so once `Layer 4` has a texture assigned, `snow_layer` cannot be added to that material. Instead, you must create a new material and then delete the old one. See the Delete a Material section below for details and cautions regarding deleting material.



The `snow_layer` texture comes with two additional snow variations that can also be painted as layers. To get these additional layers, first assign the `snow_layer` texture to `Layer 3`, then **rebuild the terrain**. If you re-select the material, the brush now shows two additional paint layers: `soft_snow` and `crust_snow`.



Soft snow is snow in which patches are partially melted. It is mostly associated with plants since plants tend to retain heat longer than the air, and plants also provide a dappled mix of sun and shade.

Crusty snow is snow in which chunks of snow have partially melted and then refrozen, so it is a mix of snow and chunks of air-filled ice. It is mostly associated with snow that has been plowed or blown off the road.

The snow types differ only in how the display engine renders the light bouncing off the snow. Unfortunately, the display engine used by the Editor doesn't model light bouncing at all, so the effect can only be seen in the game.

Bug: The Editor could assist in distinguishing the snow types, e.g. by applying a tint while a snowy texture is selected, but it does not. This means that there is absolutely no way in the Editor to tell where the different snow types have been painted. So paint with care, and try to do it only once, e.g. once everything else is nailed down.

The difference between types of snow is very subtle in the game. I arranged the below screenshot with the best lighting and camera angle that I could find, using sharp edges between the snow regions to make them more apparent. There is a slanted vertical stripe of `soft_snow` on the left and another slanted vertical stripe of `crust_snow` on the right. The normal `snow_layer` is in the center and at the edges.



Paint with `snow_layer` to put the snow texture on the ground. They optionally paint with `soft_snow` or `crust_snow` to change its appearance.

The `snow_layer` texture also supports deep snow to be piled underneath it. See the Snow Depth section on page 100 for details.

Delete a Material

You can delete a material using the context menu. When you delete a layer, the Editor makes the following changes:

- Any terrain blocks that use that material revert to using the first material instead. (This is the new first material if the previous first material was deleted.)
- The `Mask channel index` of all following layers are updated.
- The hidden bitmap that assigns materials to terrain blocks is updated with the new channel index values.

Bug: The Editor may crash when you delete a material. Save your work before deleting a material.

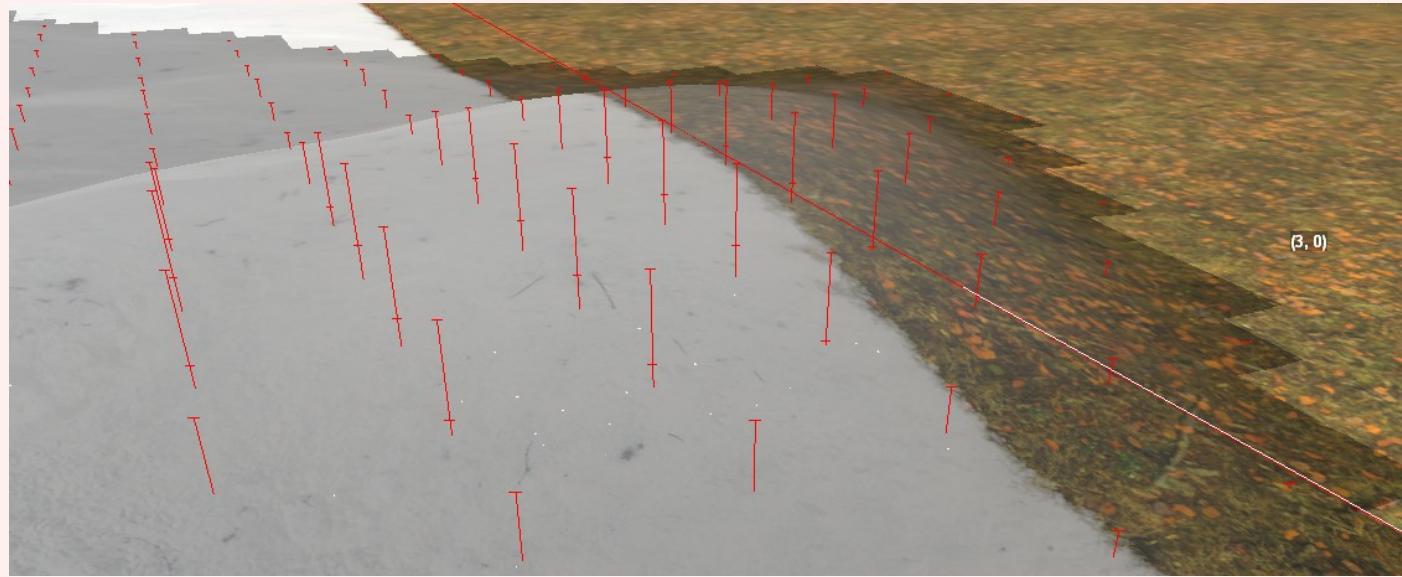
Bug: The Editor may render the terrain oddly after you delete a material. Rebuild the terrain to fix it.

Bug: The Editor is guaranteed to crash if you rebuild the terrain when there are no materials. If the map has only one material, and you want to delete it, I recommend that you first add a new material, then delete the old one.

Snow Depth

Select [\(Geometry\) → \(Snow\)](#) to paint deep snow on the terrain. By default, deep snow can only be painted where the `snow_layer` texture is present. The snow depth is ignored everywhere else.

Bug: Snow depth is ignored when it is painted elsewhere on a material that uses the `snow_layer`. However, if it is painted on a material that doesn't use `snow_layer`, it artificially raises the height of the terrain texture (as if there is snow there), but doesn't create any visual or physics effect associated with snow.



Paint snow depth by selecting `Depth` as the brush mode. This could alternatively be called snow height since the snow is piled on top of the base terrain height.

The `Value` range is -4.0 to +4.0. As with terrain height, whatever `Value` you set determines how much the snow depth changes with a single right-click and drag, regardless of how many times you drag over the same area with the mouse. You have to release the mouse button and make another pass to change the height further. As far as I can tell, a `Value` of 4.0 corresponds to about 1 meter.

Bug: Unlike anywhere else, 1.0 in the `Value` here corresponds to 20 centimeters, so +4.0 corresponds to +80 centimeters of snow depth.

While painting, the terrain is dynamically updated, but the Editor also draws thin red lines to help illustrate the depth of the snow. Tick marks are spaced every 0.5 meters along each line, starting from the top.

Although snow can be painted as deep as you want, a truck can never sink more than about 40 cm into it. To allow a truck to sink deeper, you must paint mud under the snow. If the snow is deeper than the mud plus about 40 cm, then only snow will be churned up as a truck sinks deep. But if the mud is deep enough to allow the truck to reach the base terrain, then it will start to churn up mud and soil as well.

Update Texture Layer

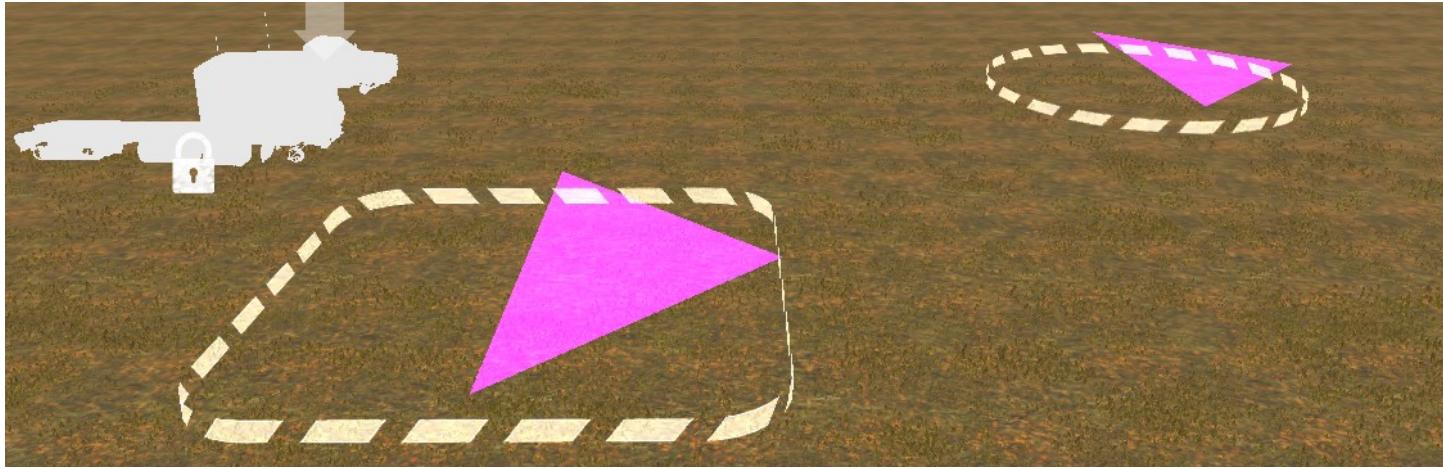
You can paint snow depth and the snow_layer texture layer simultaneously onto the terrain. To do this, enable the Update Material checkbox in the `(Snow)` brush dialog. Any snow depth you add then automatically paints `snow_layer` onto the terrain as well. Even the smallest amount of snow paints the `snow_layer` with an effective `Falloff` of zero, so it may need some touch-up afterward.

If the terrain material does not include the `snow_layer` texture in `Layer 2`, `Update Material` has no effect.

Bug: If you click `Undo Current Changes` in the brush dialog, it undoes the snow depth changes, but it doesn't undo the changes to the `snow_layer` material.

Zones

Zones define regions on the map that trigger various effects. E.g. they are used for gaining or completing an objective, for visiting a garage, and for triggering sound effects. Each zone is surrounded with a dotted line in the Editor and the game. In the game, the dotted line is animated to slide around the perimeter of the zone. In the Editor, the dotted line is static, but the Editor draws an additional pink triangle to point in the direction of the zone's orientation. The orientation of the zone makes a difference for certain zone types.



Zones are commonly associated with 3-D models such as garages, fuel pumps, etc., but the model is just set dressing. Only the zone has the associated game logic.

Properties that are the same for all zone types are described in this section. Properties that are manipulated by the Zone Settings Editor and are described in the following sections.

ID

Each zone must have a unique `Id` property value that links it to its separate type-specific properties in the Zone Settings Editor. I have found that the `Id` can be almost any text that doesn't include a `"`, but Saber strongly recommends sticking to a combination of lowercase `a - z`, `0 - 9`, and `_`, with no spaces.

Name and Icons

The `Name` property specifies the name that is displayed for the zone on the navigation map. Unlike the `Id` property above, the `Name` does not need to be unique. If no `Name` is specified, the `Id` is used instead.

The `Name` property isn't displayed while the player is driving. E.g. if the player unlocks a garage named "Fred's Garage", the game only displays a generic "Garage Unlocked" message.

The `Name` property can be localized. See the Localization section on page 148 for details.

The `Icon 40x40` property specifies the name of an icon that is displayed above the zone when the player drives close to it. The same icon is also displayed on the navigation map.

The `Icon 30x30` property specifies the name of an icon that is displayed in the list of locations on the left side of the navigation map.



Unfortunately, there is not a convenient menu of icon images to choose from the Editor. Refer to the Zone Icons section on page 165 for a table of icons and their corresponding names. Be sure to copy each name exactly as it appears in the table, even if it looks like the name has a typo. If an icon name is specified incorrectly, the game will display a small red and yellow placeholder image (☒) instead.

If a 40×40 icon is used in place of a 30×30 icon or vice versa, it will look generally OK, but it won't be properly centered.

If an icon property value is left blank, then no icon is drawn. However, other cues are present as usual to indicate the presence of the zone. The screenshot below shows two fuel stations on the map, one with associated icons and one without.



The `Is Visible On Minimap` property defaults to `True`. If you change it to false, then the zone is not displayed on the navigation map or on the list to the left of the map. In the case, the `Name` property doesn't

matter because it is never used. The zone boundary still appears during normal driving when appropriate, of course. However, if `Is Visible On Minimap` is `False`, the icon above the zone is also suppressed.

Bug: The SnowRunner Editor and its documentation consistently misuse the term “minimap” when they mean the full-size navigation map. SnowRunner does not have a true minimap.

SnowRunner automatically suppresses any display of the zone in most circumstances where doing so is appropriate (e.g. when a zone is only used for an inactive objective). It also automatically suppresses the zone from the minimap for certain zone types:

- Garage exit.
- Recovery zone.
- Manual cargo loading zone.

Since the game already suppresses the zone when appropriate, there is usually no need to set `Is Visible On Minimap` unless you want to make life difficult for the player. However, hiding the zone on the map is useful for the interaction zone for a cargo delivery with manual unloading, described on page 133.

Zone Perimeter

The `Shape Type` property determines the shape of the zone boundary: either a rectangle (with rounded corners) or a circle.

When the `Shape Type` is `Rectangle`, the length of the sides parallel to the zone’s direction are set by the `Length or diameter` property, measured in meters. The length of the perpendicular sides are set by the `Width` property values (in meters). The radius of its rounded corners is determined by the `Rounding radius` value (in meters).

When the `Shape Type` is `Circle`, the diameter of the circle is set by the `Length or diameter` property value (in meters). The `Width` property is ignored.

Bug: When the `Shape Type` is `Circle`, the game ignores the `Width` value. However, the Editor still draws the zone’s front-facing arrow using the `Width` as its width. For better visuals in the editor, you might want to set the `Width` equal to the `diameter` for a `Circle` zone.

Ribbon Shape and Size

A ribbon of dotted lines is animated around the perimeter of the zone. By default, the `Is wall` property is `False`, and the ribbon lies flat against the ground. If the `Is wall` property is set to `True`, the ribbon is vertical.



The width of the ribbon is determined by the `Thickness` property, but it is not measured in meters. Instead, the ribbon width is the `Thickness` value \times 0.5 meters.

Vertical Ribbon

When `Is wall` is `True`, the layout of the ribbon in the top-down view is determined by the `Length or diameter`, `Width`, and `Rounding radius` properties as described above. Independently, the layout of the ribbon in the side view is determined by the `Thickness` and `Height` properties.



The `Rounding radius` isn't constrained by the `Thickness` of the ribbon, so the minimum `Rounding radius` is 0, entirely removing the rounding from the corners.

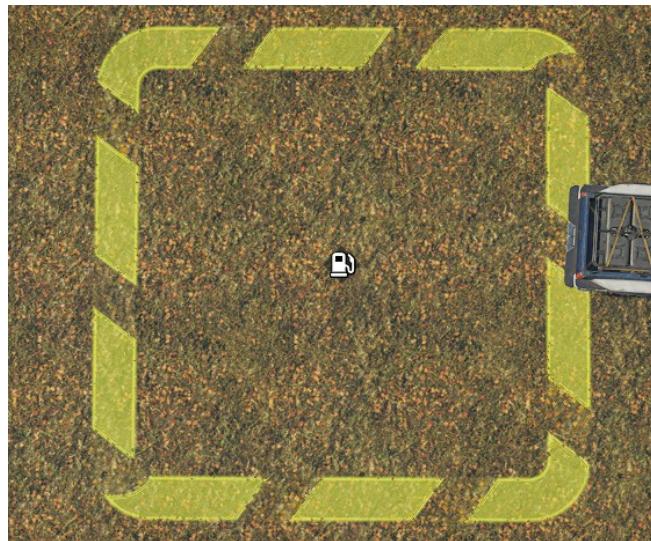
When the `Height` value is 0, the vertical ribbon is centered at ground level. Its height is adjusted by the `Height` value, measured in meters.

Bug: The Editor places the forward-facing pink arrow (visible only in the Editor) at **half** the height given by the `Height` value.

To raise the bottom of the ribbon to exactly ground level, the `Height` must be set to half the ribbon width. However, since `Height` is measured in meters but `Thickness` is measured in half meters, the `Height` value must be one-quarter of the `Thickness` value.

Flat Ribbon

When `Is wall` is `True`, the layout of the ribbon in the top-down view is determined by a combination of its `Length or diameter`, `Width`, `Rounding radius`, and `Thickness` properties.



Bug: Although the game draws the ribbon either flat or vertical, depending on the `Is wall` property, the Editor always draws the ribbon vertically. The Editor still ignores the `Height` property when placing the ribbon, however, instead drawing the bottom of the ribbon above the ground by half its width. On the other hand, the Editor uses the `Height` property to set the height of the forward-facing arrow.

This makes it much harder to experiment with different property values to achieve the desired look. So the calculations below that determine the exact ribbon placement may be especially useful to you if you don't mind the math.

When the ribbon is flat on the ground, it has additional invisible padding equal to half of the ribbon width on each side. Since the ribbon width itself is half the `Thickness` value in meters, the total padded ribbon width is

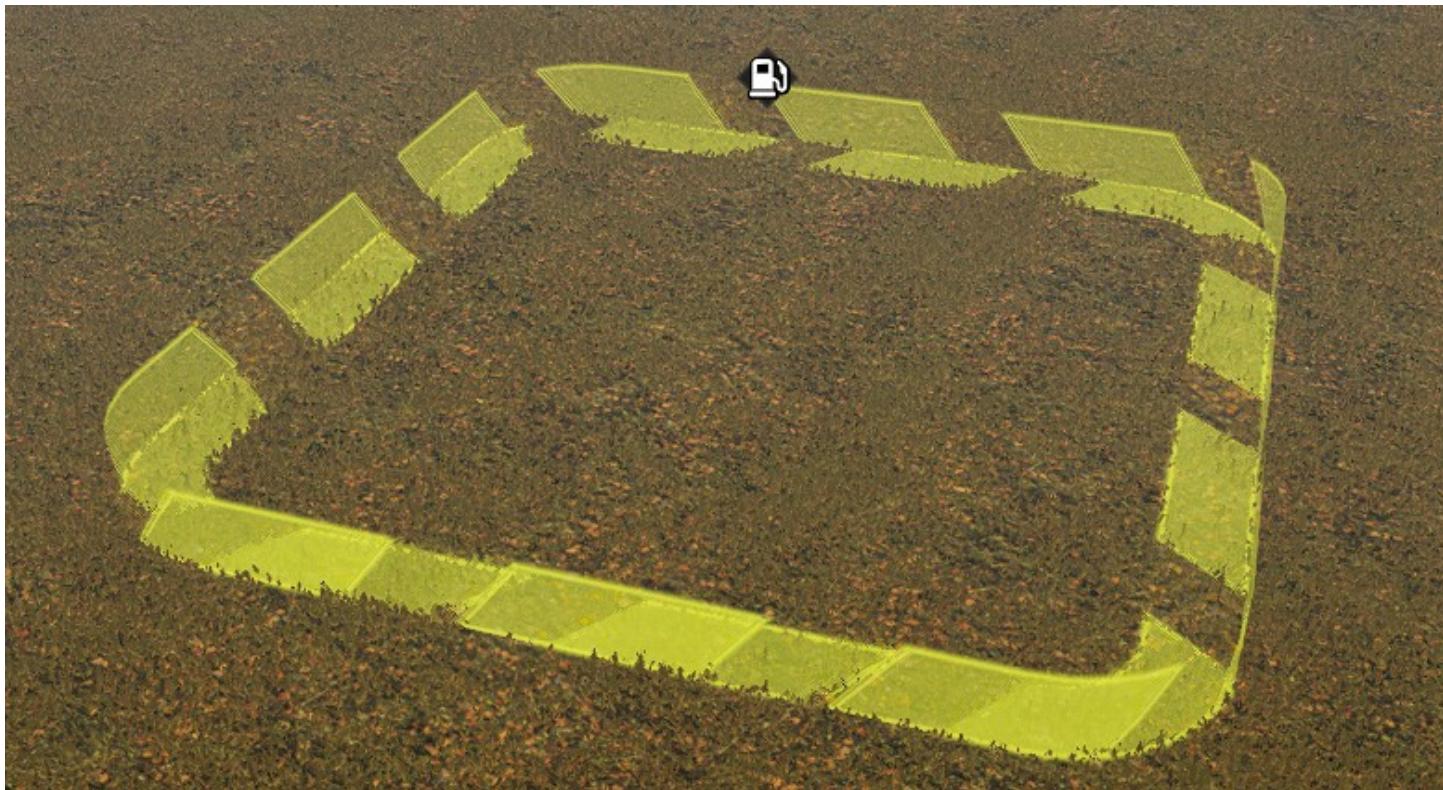
exactly the **Thickness** value in meters. This padded ribbon width appears to be what SnowRunner uses for its internal calculations, and it is important when calculating the precise placement of the ribbon.

The ribbon is placed so that the outer edge of its **padded** width aligns with the dimensions provided by **Length or diameter** and **Width**. This means that the ribbon itself is inset by half its width.

The **Rounding radius** sets the **outer** radius of the **padded** ribbon width, in meters. The inner edge of the ribbon necessarily has a smaller radius. To avoid a negative radius, the game automatically reduces the **padded** ribbon width to no greater than the **Rounding radius**. This means that the visible ribbon width is no more than half the **Rounding radius**, and the inner edge of the visible ribbon is always rounded, never a sharp corner.

The actual detection area for the zone is determined by the outer edge of the padded ribbon width, which is exactly set by the **Length or diameter**, **Width**, and **Rounding radius** properties. However, you might prefer to set the visible part of the ribbon to particular dimensions. If so, use the calculations below:

- **Length** = desired visible outer length + $0.5 \times \text{Thickness}$
- **Width** = desired visible outer width + $0.5 \times \text{Thickness}$
- **Rounding radius** = desired visible outer radius + $0.25 \times \text{Thickness}$



Other Ribbon Properties

The speed of ribbon animation is given by the `Scroll speed and direction` property, in clockwise meters per second. The animation is counter-clockwise if the value is negative. If `Is wall` is `False`, the speed is measured at the outer edge of the padded ribbon.

The brightness of the ribbon is determined by the `Opacity` property. Since the ribbon color is added to the ground color, it is never truly opaque, but larger values make it appear more opaque, and smaller values make it appear more translucent.

Bug: The `Opacity` property only affects how the ribbon looks in the Editor. In the game, the ribbon's color and brightness are determined by the zone type, and the `Opacity` property is ignored.

Hilly Zones

If the terrain in a zone is not flat, the zone boundary gets distorted. If `Is wall` is `False`, the ribbon conforms to the terrain surface. If `Is wall` is `True`, the ribbon makes travels follows a straight line from the height at one corner to the height at the next corner (modulo the `Rounding radius`).

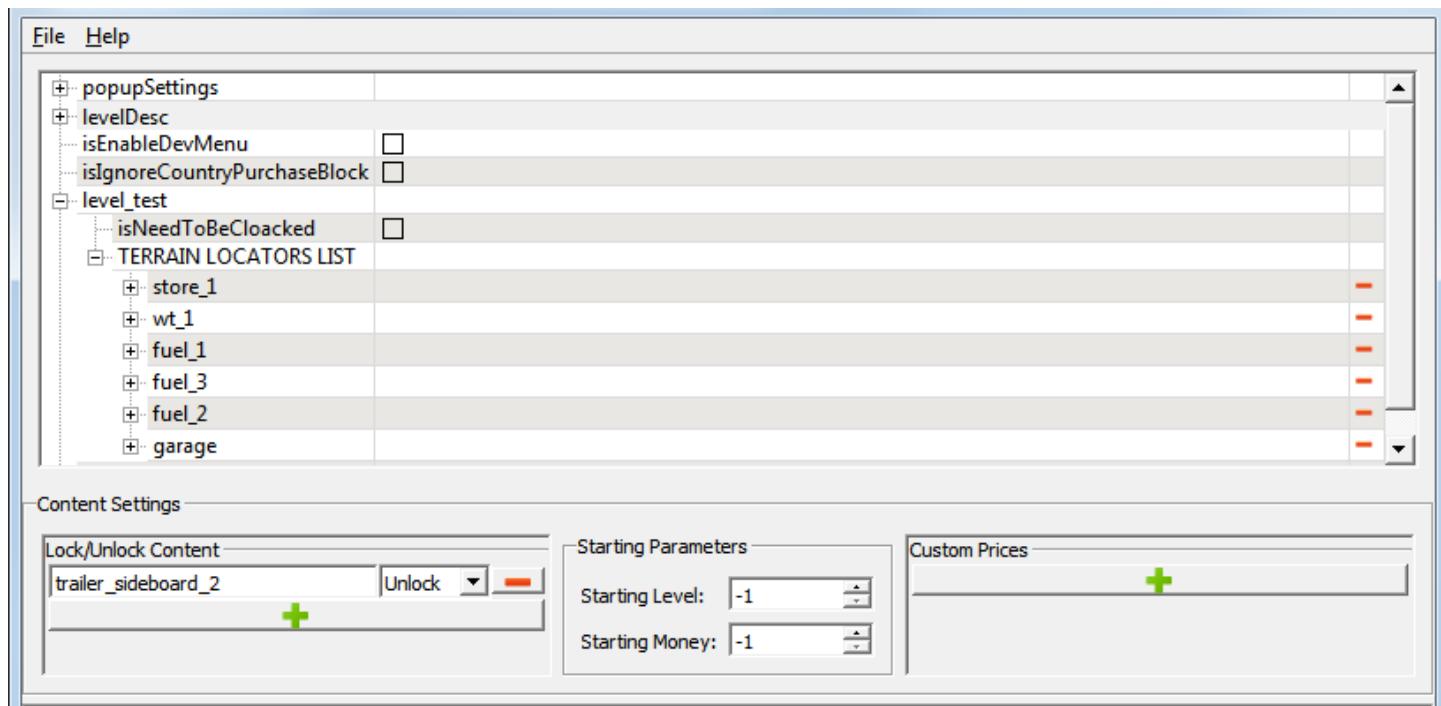


Zone Settings Editor

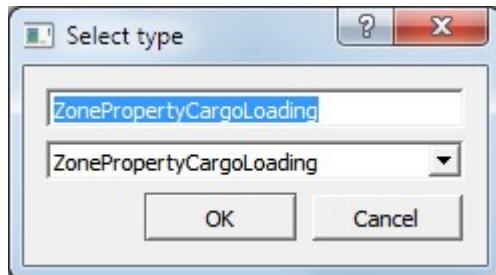
The Zone Settings Editor allows you to set the type of each zone and the various properties that are specific to the zone type. For unknown reasons, it also includes a random assortment of other non-zone-related properties.

To start the Zone Settings Editor from the main SnowRunner Editor, click the `Zone settings` button on the toolbar (the second button from the right). **Caution:** while the Zone Settings Editor is open, the main SnowRunner Editor window is completely unresponsive. Close the Zone Settings Editor to return to the main SnowRunner Editor.

This section describes only the zone properties. In the hierarchical list, open the container for your level name, and then the subcontainer `TERRAIN LOCATORS LIST`. This list is automatically populated with the IDs for each zone that you set in the main Editor. If a zone does not have an ID, it is not listed.



To add properties to a zone, open its container, then click the green plus to the right of `props`. Then select the desired zone type from the dropdown menu in the dialog box and click `OK`.



Confusingly, the dialog box has two editable fields, and selecting an item from the dropdown menu updates both of them. Details TBD.

The zone type properties are weirdly named. The Zone Types section on page 115 describes the various zone types and the corresponding property. Depending on the zone type, the Zone Settings Editor may make a number of subproperties available.

Manage the Zone ID

The loose connection between the zone `Id` in the main SnowRunner Editor and the zone properties in the Zone Settings Editor is a conceptual minefield. Although the two Editors attempt to keep your changes in sync, it is easy to lose confidence in what they are doing.

Add or Delete a Zone

The main Editor saves most of its property data in `Media/prebuild/<level_name>.xml`, while the Zone Settings Editor saves the zone properties in `Media/levels/<level_name>/zone_settings.json`.

If you add or delete a zone in the main Editor, it doesn't touch the `zone_settings.json` file directly. Instead, when you start the Zone Settings Editor, the main Editor tells it what zones currently exist. Thus, the Zone Settings Editor allows you to edit the same zones that are present in the main Editor.

Note that if you don't select `Save` in the Zone Settings Editor, no changes are made to the `zone_settings.json` file.

Tip: If you delete a zone in the main Editor the zone and its properties won't be shown in the Zone Settings Editor. However, as long as you select `Save` in the Zone Settings Editor, the zone settings can be recovered. Close the Zone Settings Editor (without saving), re-create the zone in the main Editor, restart the Zone Settings Editor, and the zone's properties should be there once more.

Rename a Zone ID in the Main Editor

If you change the `Id` of a zone in the main Editor, the Editor keeps track of this rename. The next time you start the Zone Settings Editor, the main Editor tells it to change the name(s) in the `zone_settings.json` file before it opens the window where you can edit the zone settings.

This is weird, so let me say it again. If you renamed a zone `Id` in the main Editor, the `zone_settings.json` file can change even though you never selected `Save` in the Zone Settings Editor.

This oddity only applies to renamed zone IDs. If there are also new or deleted zones, those still aren't updated in `zone_settings.json` until you select `Save` in the Zone Settings Editor.

Note that if you then close your map in the main Editor without saving, your zone IDs will be inconsistent between the `<level_name>.xml` file and the `zone_settings.json` file.

This inconsistency between files doesn't happen in the other direction, however. If you save your work in the main Editor while it is tracking a renamed zone ID, the main Editor quietly starts the Zone Settings Editor in the background and tells it to change the name in the `zone_settings.json` file so that both files are in sync.

Rename a Zone ID in the Zone Settings Editor

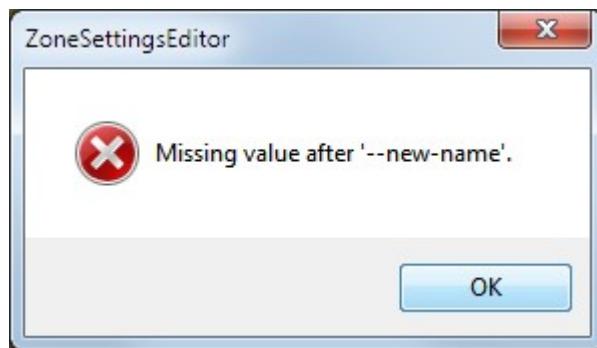
If you change the ID of a zone in the Zone Settings Editor, it does not attempt to make a corresponding change in the main Editor. If you save your change in the Zone Settings Editor, the two files will be out of sync.

After closing the Zone Settings Editor, you can immediately make the same change in the main Editor. This will attempt to make a corresponding rename in the `zone_settings.json` file, but since the old ID no longer exists in that file, it is harmless.

After you manually change and save both files in this order, the two are correctly in sync. However, it's easier to rename the zone ID in the main Editor first and let it deal with synchronizing the change.

Clear the Zone ID

If you clear the `Id` of an existing zone, the main Editor tracks this as a rename event. The next time you save your changes in the main Editor or start the `Zone Settings Editor`, you'll get an error that a name is missing.



This is harmless. All of the other zones are presented correctly, including any that have been renamed to valid IDs.

Uncommitted Id Edits

When you type a new name into a zone's `Id` property in the main editor, the change is not committed right away. This uncommitted status can be seen in the `Scene View`, which doesn't update the zone name even when you press return. To commit your edit, left click in the main panel or click something else in the `Scene View`.

If you create a new zone, assign it an `Id`, and then start the Zone Settings Editor while the `Id` change is uncommitted, then the Zone Settings Editor won't know the new zone's ID, and it won't include it in the list of zones.

Bug: After you close the Zone Settings Editor, the main Editor will no longer have the zone selected, but it still won't have committed the `Id` change. Select the zone again, then click elsewhere to deselect it, and it should finally be committed. You can now restart the Zone Settings Editor to see and edit the new zone.

Things are worse if you have an existing zone in the `zone_settings.json` file, and you rename its `Id` in the main Editor without committing the change. If you then start the Zone Settings Editor, the main Editor tells it to rename the zone according to the uncommitted change, but the main Editor **also** says that the current list of zones includes the zone with its old ID.

Bug: Oh yeah, that's a bug.

As an example, assume you have an uncommitted ID change from `x` to `y`, and you start the Zone Settings Editor. the ID is changed in `zone_settings.json` file from `x` to `y`. However, the Zone Settings Editor is also told that the list of zones includes `x` and not `y`. Since the Zone Settings Editor no longer has any properties for zone ID `x`, it lists `x` without any properties. If you select `Save` now, you'll lose your properties for the zone. If you realize the situation, close the Zone Settings Editor without saving. Select the zone again the main Editor, then click elsewhere to commit the change, then restart the Zone Settings Editor.

Tip: Make a habit of clicking elsewhere to commit a change to a zone's ID before doing anything else.

Duplicate IDs

Now that you have some understanding of how ID changes are synchronized, it is apparent how confusing it is to ever have two zones with the same ID.

Bug: If you change the `Id` of a zone multiple times in the main Editor, the Editor tracks this as a series of renames, not as a single change from the original ID to the newest ID. If any of the intermediate IDs were a duplicate of another zone's ID, this will cause problems even though the Zone Settings Editor was never invoked while duplicate names were present.

Of course, if you choose `Duplicate` on a zone, then the new zone has the same ID. You could avoid using `Duplicate`, but it can be convenient for duplicating zone locator properties or aligning zones near each other.

If you use `Duplicate` to make one zone into multiple, whichever one you rename first ends up with the original zone's properties. Although the main Editor presents the other renames to the Zone Settings Editor, there is no longer a zone with that ID to rename.

Tip: You can actually take advantage of the above bug. After duplicating a zone, rename the original zone, then rename the duplicates, then rename the original zone back to its original name. When you start the Zone Settings Editor, the original zone will have its original name **and** its original properties.

Zones Used by Objectives

Some properties under the `objectiveSettings` category in the Zone Settings Editor take a zone `Id` as a value. However, the Editors make no attempt to keep these IDs in sync as you make changes. If you change the zone ID in one place, it's up to you to also change it in the other place.

Zone Settings Editor Controls

The hierarchical list in the Zone Settings Editor is similar to the ones in the main Editor, but some of the controls are different. The complete list of confusing, context-sensitive Zone Editor Controls is below. Controls that are useful in all contexts are **highlighted**.

Some mouse controls can be performed with either left or right click:

- Click on a **+** to the left of a hierarchical container to expand that container, showing its contents.
- Click on a **-** to the left of a hierarchical container to contract that container, hiding its contents.
- Double click on an editable string to edit it. Property values are generally editable, as are zone IDs and `ZoneProperty` names.
- Double click on a non-editable container to expand or contract a container.

Others can only be performed with a left click:

- **Left click on a green + to the right of a property to add a sub-property within it.**
- **Left click on a red - to the right of a property to delete it and all of its contents.**

Certain actions can also be performed using the keyboard:

- The up and down arrows move the selection up and down the list.
- The right arrow does different things in different contexts:
 - On an unexpanded folder or container, it expands the container.
 - On an expanded folder or container, it moves to the first item in the container.
 - On a non-container item, it does nothing.
- The left arrow does different things in different contexts:
 - On an expanded folder or container, it hides the items in the container.
 - On an unexpanded folder or container, it moves to the container of the current item (up one level in the hierarchy).
 - On a non-container item, it does nothing.
- F2 edits the selected item. This can only edit its name if the item was selected via the keyboard. It can edit the value if you selected the item by clicking in the value field.
- The **Home** key moves the selection to the top of the list.
- The **End** key moves the selection to the bottom of the list.
- The **Page Down** key moves the selection down by approximately one page.
- The **Page Up** key moves the selection up by approximately one page.
- **Ctrl-S** saves changes.

Bug: A change isn't saved if it hasn't been committed. If the cursor is still visible in a text field, the edits to that field aren't yet committed. Commit your changes by pressing **Enter** or selecting another item.

- **Ctrl-Q** exits the Zone Settings Editor.

Bug: If the only change hasn't yet been committed, the Zone Settings Editor won't prompt you to save before quitting.

In all cases, the panel automatically scrolls to keep the selected item in view.

Zone Types

Objective delivery zones are not assigned a property here. They are handled separately as described in TBD.

Garage Entrance

A zone with the property `ZonePropertyGarageEntrance` allows the player to enter a garage. The zone boundary is yellow. The forward orientation of the zone only matters if the zone is also used as a garage exit.

Each map may contain only one garage, and Saber says that a map “can” contain only one garage entrance. However, my testing shows that a map can contain multiple garage entrances, although they all enter the same garage. Taking advantage of this may be risky, however. Also, while the player is in the garage, the building list next to the navigation map shows the player icon in both of the garage entrances.



If a separate garage exit is not specified, then the garage entrance also acts as the garage exit. In this case, the zone’s orientation is important, as described in the following section.

Garage Exit

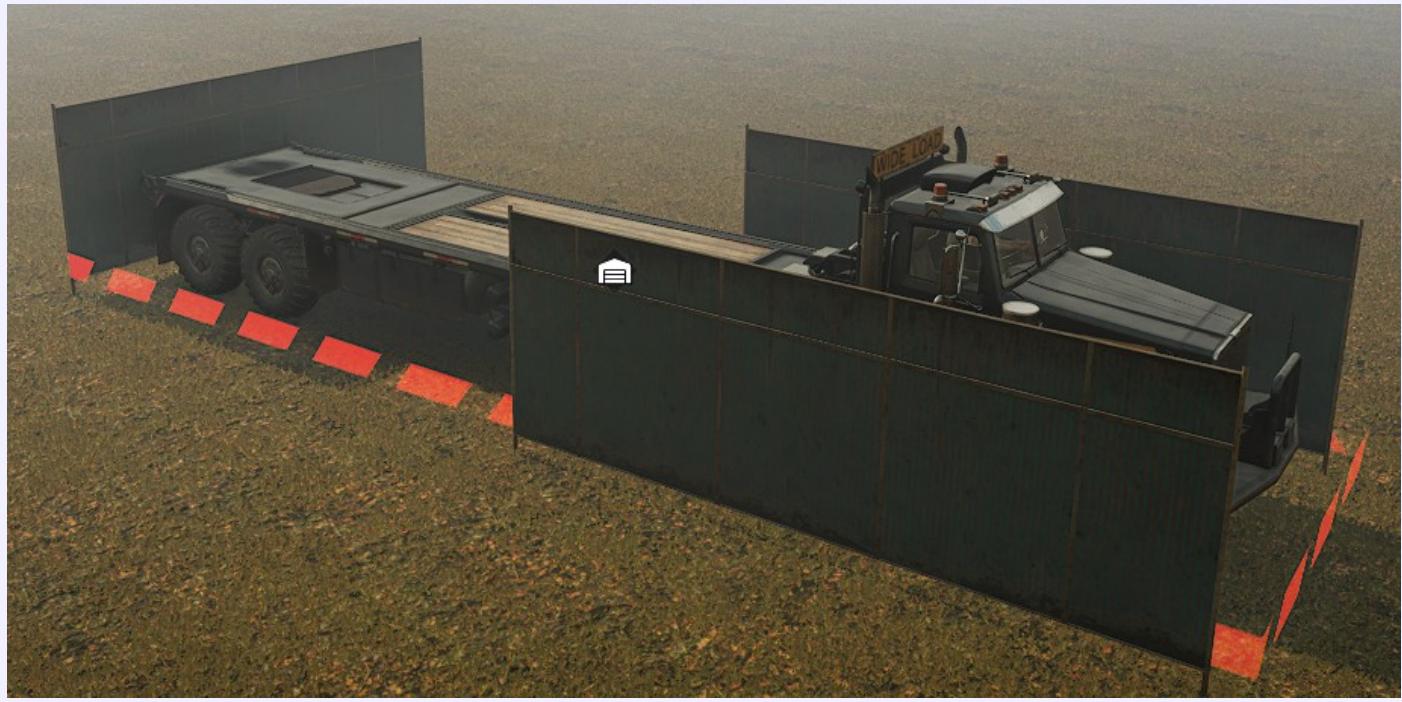
When the player exits the garage, the truck is placed in a zone with the property `ZonePropertyGarageExit`. If there is no such zone, then the truck is placed in a zone with the property `ZonePropertyGarageEntrance` instead.

A `ZonePropertyGarageExit` zone has a red boundary. It is automatically suppressed from appearing on the navigation map or building list, regardless of the `Is Visible On MiniMap` property value.

Tip: Clear any icons from `Icon 30x30` and `Icon 40x40` to prevent a stray icon from appearing over the zone in the driving view.

The truck is centered in the exit zone, facing in the same direction as the zone’s orientation. To avoid a catastrophe, make sure that the largest truck can exit without intersecting an obstacle. If you make a zone of the corresponding dimensions, it will not only help you keep the area clear of stray models, but it will also help guide the player to keep the area clear of parked trucks.

Tip: My measurements indicate that a zone with a `Length` of 17 meters and a `Width` of 5 meters will accomodate the longest and widest trucks (including the year 1 DLC). However, extrapolating from Saber's recommendations, 21 meters by 9 meters may be safer for future DLC and player mods.



Saber says that a map “can” contain only one garage exit. My testing shows that a map can contain multiple garage exits, but the game chooses one of them randomly for where to respawn the truck. The random choice is made when you pack the map, so you can’t get the fun of randomizing the truck’s position each time they exit the garage, either.

Recovery Zone

When the player chooses `Recover` from the `Functions` menu, the truck is recovered to the garage if one has been discovered. Otherwise, the truck is recovered to the zone with the `ZonePropertyRecover` property.

No zone boundary appears for a `ZonePropertyRecover` zone, and no icon appears in the driving view. The zone is also automatically suppressed from appearing on the navigation map or building list, regardless of the `Is Visible On MiniMap` property value.

The recovery zone should be placed to avoid obstacles the same as for the garage exit, above.

Tip: Place the recovery zone where the active truck starts so that the player returns to where she started rather than warping to somewhere new.

If the garage is not yet discovered, and there is no zone with the `ZonePropertyRecover` property, then recovery fails. It's not nice to do this to the player.

WARNING!

Evacuation is not available. The garage is not found and evacuation area is not established.

Trailer Store

A zone with the property `ZonePropertyTrailerAttach` allows the player to buy or sell a trailer attached to the truck. The zone boundary is yellow. The forward orientation of the zone doesn't matter.

When the player buys a trailer, it doesn't need to fit within the zone, but the purchase fails if the trailer would intersect an obstacle. Make sure there is sufficient room in at least one direction for a very long trailer such as the superheavy semi-trailer (which adds about 20 meters to the truck length).

Tip: There should usually be plenty of room in the direction of the store entrance for a long trailer. However, the player would often prefer to drive forward out of the trailer store after purchasing a new trailer, so consider leaving enough room for the trailer in the direction opposite the store entrance.

Fuel Station

A zone with the property `ZonePropertyFuelStation` allows the player to refuel her truck. The zone boundary is yellow. The forward orientation of the zone doesn't matter.

Service Hub

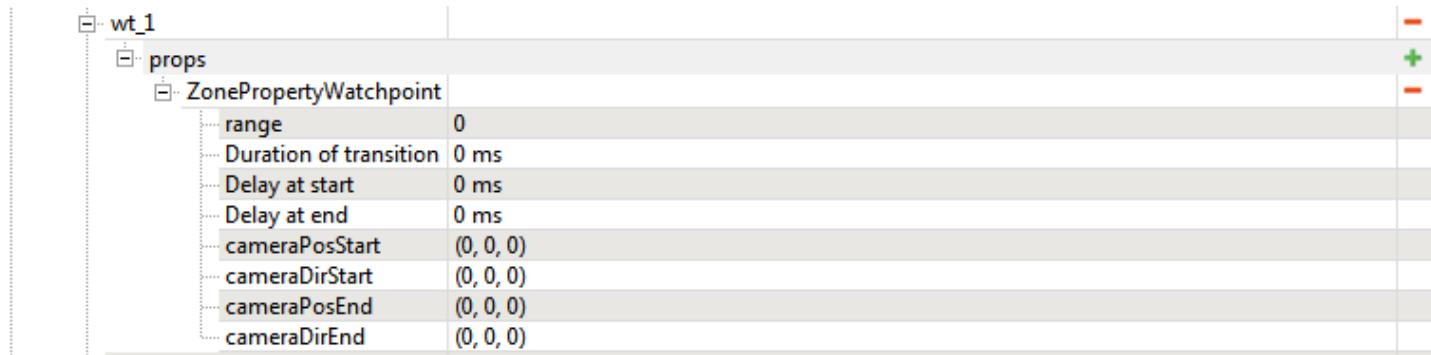
A zone with the property `ZonePropertyFuelStation` repairs the player's truck. The zone boundary is yellow. The forward orientation of the zone doesn't matter.

Unlike the fuel station, the service hub doesn't cost money, so it happens automatically without confirmation.

Watchpoint

A zone with the property `ZonePropertyWatchpoint` reveals the surrounding region on the map. The zone boundary is blue. The forward orientation of the zone doesn't matter.

Unlike the above zone types, extra properties are added below `ZonePropertyWatchpoint` as soon as it is added.



The `range` property specifies the radius that is revealed on the map, in meters. See page 153 for more information about revealing the map. Areas within 39 meters of the active truck are revealed without the aid of a watchpoint, so the watchpoint range should generally be significantly larger than that.

The remaining properties specify the camera behavior if the player chooses to [Launch Observation](#):

- `cameraPosStart` specifies where the camera starts at the beginning of the observation animation.
- `cameraDirStart` specifies the camera orientation at the beginning of the observation animation.
- `Delay at start` specifies how long the camera pauses before starting to move.
- `Duration of transition` specifies how long the camera takes to move to the end position.
- `cameraPosStart` specifies where the camera stops at the end of the observation animation.
- `cameraDirStart` specifies the camera orientation at the end of the observation animation.
- `Delay at end` specifies how long the camera pauses after moving and before the observation ends.

When editing the times, the “ms” string is uneditable. The time is always in milliseconds (thousandths of a second), so e.g. `7000 ms` is 7 seconds.

When editing the camera positions and orientations, the three values split into separate `x`, `y`, and `z` fields, each of which can be edited.

Tip: The easiest way get a good camera position and orientation is to position the camera in the Editor, then enable the [Statistics](#) toggle button in the toolbar. (Remember to close the Zone Settings Editor first.) The statistics list the camera position followed by the direction: $(x; y; z) > (x; y; z)$.

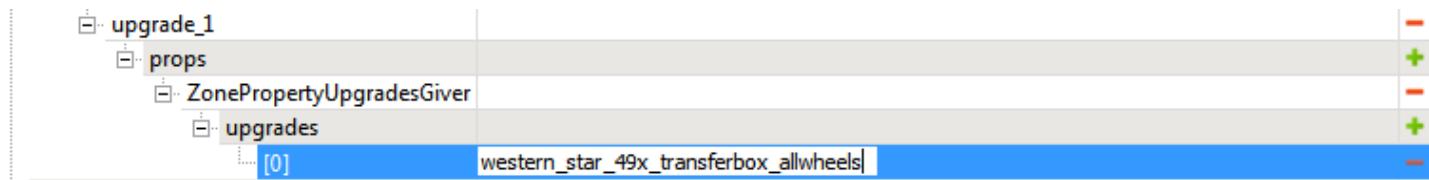
Camera: (-26.68; 30.19; -25.96) > (0.88; -0.24; 0.40)

Free Upgrades

A zone with the property `ZonePropertyUpgradesGiver` grants the player with a free part and the ability to buy more. The zone boundary is yellow. The forward orientation of the zone doesn't matter. The zone disappears after being used.

`ZonePropertyUpgradesGiver` comes with an `upgrades` subcategory, to which any number of numbered properties can be added. Click the green `+` to the right of `upgrades` to add a numbered property below it.

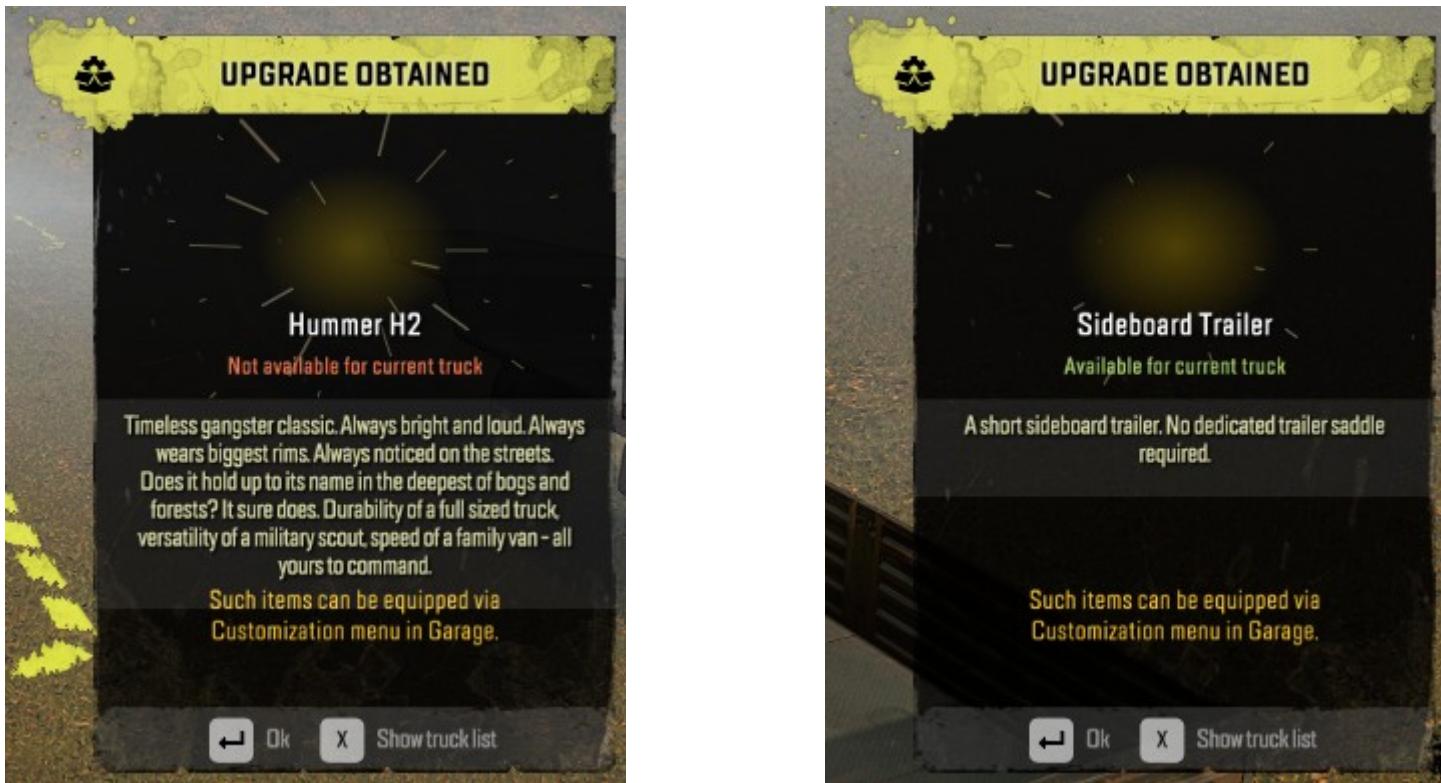
Enter an add-on name in the `[0]` property to cause the add-on to be unlocked when the player reaches the zone. The player is also gifted with a free part, which is then available to be installed the next time she visits a garage. If the add-on is already unlocked, then the upgrade still grants the player a free one, even if she already owns one or more.



Bug: `ZonePropertyUpgradesGiver` implies plural upgrades, and any number of numbered properties can be added within the `upgrades` subcategory, but the game grants only the first upgrade to the player. Everything after `[0]` is ignored.

Upgrades can also be awarded for completing objectives. See TBD.

You can name a truck type or trailer type in the `upgrades` property, and the game will unlock it (but not grant a free copy). However, this feature is not documented and displays incorrect messages to the player with a missing icon, so it clearly isn't intended to be used. The normal way to unlock a truck or trailer is to find it on the map.



The various `saddle_low` and `saddle_high` addons clearly also aren't intended to be locked or upgraded. A `saddle_low` simply cannot be locked at all. A `saddle_high` can be locked, but it is still supplied for free when a trailer that requires it is purchased from the trailer store. TBD: move this to the 'lock parts' section. Add similar text to the 'adjust prices' section.

If `upgrades→[0]` has an invalid part name, then the game appears to grant the part to the player, although there is no way to buy or install the part.

If no upgrades are valid, the zone is displayed in the game, but the player cannot activate it.

Automatic Cargo Loading Zone

A zone with the property `ZonePropertyCargoLoading` allows the player to spawn packed cargo directly onto a truck or attached trailer. The zone boundary is yellow. The forward orientation of the zone doesn't matter.

`ZonePropertyCargoLoading` comes with a subproperty `platformId`. Enter the `Id` of a manual loading zone to associate it with the automatic loading zone. See the next section for details.

`ZonePropertyCargoLoading` also comes with a subcategory `cargoSettings`. Click the green `+` to the right of `cargoSettings` to add a numbered category below it. Each numbered category adds a cargo type that can be loaded in the zone.

The `name` subproperty specifies a cargo type. When editing this property value, you can either type in a name or select a name from the dropdown menu. Note that the menu is missing some names, so you'll need to type those in by hand. Reference information for cargo types is on page 171. If a `name` is mistyped, that cargo type is ignored.

The `Amount` subproperty specifies how much of that cargo is available from that loading zone (or the associated manual loading zone). If the `Amount` is `-1`, an unlimited quantity is available. If the `Amount` is `0`, the cargo is shown to the player but with the caveat "No more cargo left."

Tip: If there isn't another source for goods with a limited quantity, player mistakes can potentially ruin her game. E.g. if the player spawns goods that aren't needed yet, her only option is to use a crane to set them aside for later. If a crane isn't available or the player chooses instead to destroy the goods, then a later objective may be undeliverable. On the other hand, if there is a limited source for goods nearby and an unlimited source further away, the punishment for a mistake is the usual one for SnowRunner: more driving.

The available cargo types are presented to the player in a random order. If the same cargo type is listed twice, only one of them is used; their amounts are not cumulated. If no cargo types are valid, the zone is not displayed in the game.

If the automatic loading zone is enabled on the navigation map, it helpfully shows which cargo types are available at the zone. However, the map can only list 5 cargo types (chosen at random). If there are more than 5 cargo types, the map tells you how many more are available.



Manual Cargo Loading Zone

A zone with the property `ZonePropertyManualLoading` allows the player to spawn unpacked cargo directly onto the ground. The player can then use a crane to manually load the cargo onto a truck. The zone boundary is red. The forward orientation of the zone matters.

The manual cargo loading zone must have the `ZonePropertyManualLoading` property and must be associated with an automatic cargo loading zone. Enter the manual zone's `Id` in the `platformId` property of the automatic zone to associate them. The player can then enter the automatic loading zone and choose whether to spawn goods directly onto the truck or into the manual loading zone.

If a cargo type has limited quantity in the automatic loading zone, then the available quantity is reduced whether the goods are spawned to the automatic or manual loading zone.

Since the manual cargo loading zone is effectively controlled by the paired automatic cargo loading zone, it has no additional properties of its own.

If we assign a “forward” direction for cargo based on how it is loaded on the truck, then the cargo spawns in the manual loading zone pointing 90° to the right of the zone orientation. I.e. long goods are oriented with the long end perpendicular to the zone’s direction.

The game attempts to spawn cargo just above ground level. If the ground is hilly, it tries to spawn the cargo high enough to not spawn inside the ground. If it detects a model in the way, it spawns cargo just above the model. Its heuristic isn’t very good, though. It does best when the ground is very flat and any model underneath is well centered and also flat (such as the various `platform` models).

Tip: To perfectly center a model in the zone, simply copy the zone’s `Org X` and `Org Z` values to the model’s `X` and `Z` properties. Unfortunately, the zone’s direction isn’t even in the same format as the model’s, so you’re on your own there. The default orientation for zones (pointing along the X axis) happens to match the default orientation for platform models, however, so you’re all set if that orientation works for you.

The game’s spawn height heuristic performs poorly with the standard log trestle (`logs_scavange_02`), but it compensates by spawning logs 2 meters higher to give them room to drop into place.

The zone’s `Height` property has no influence on the cargo spawn location. However, if `Is Wall` is `True`, the `Height` property is useful to position the zone ribbon just above a platform model.

The manual loading zone always has a “hook” icon above it in the driving view, regardless of any icons assigned to the zone properties. The zone is automatically suppressed from appearing on the navigation map or building list, regardless of the `Is Visible On MiniMap` property value.



Tip: The zone's dimensions have no effect on how the cargo is spawned. But setting the dimensions to the maximum size of potential cargo helps inform the player where to expect the cargo to appear and where to avoid parking a truck.

Crafting Zone

- `ZonePropertyStorehouseCraft`: TBD.
- `ZonePropertyEnergy`: TBD.

Multiple Zone Types

Multiple types can be assigned to a single zone. For zone types that do something automatically, all automatic effects are invoked. For other zone types, the player can switch among the various interaction options.



Tip: With no way to signal the multiple zone types on the map, the player might not notice that they can switch to another interaction. The main campaign uses this feature only when the additional option is a objective delivery zone, which has its own map indicator.

Objectives

Because a SnowRunner objective usually involves visiting multiple zones, objectives are not specified within a zone. Instead, objectives get a separate `objectiveSettings` category in the Zone Settings Editor. The objective properties are also saved in a separate file, `objective_settings.json`.

An objective can take one of three forms:

- A contract is activated automatically when unlocked, without the player needing to visit a particular zone. Contracts are organized within the game based on employer offering the contract.
- A task is offered when the player visits a particular zone.
- A contest is also offered when the player visits a particular zone, and it offers different rewards depending on the speed of completion.

All three objective types are very similar. Any differences are discussed where appropriate in the sections below.

Objective Status Terminology

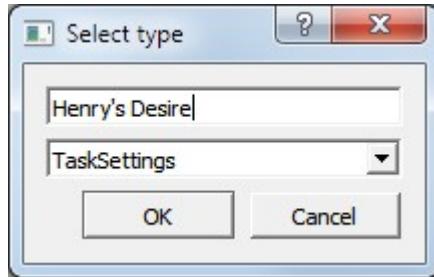
I use the following terms to describe the status of objectives:

- Locked: The player does not have the required rank to activate the objective, or it is blocked by another objective.
- Unlocked: The player meets all of the requirements for an objective to be offered or activated. An unlocked task or contest is offered when the player drives to the starting zone. An unlocked contract automatically moves to the Activated state.
- Offered: The game offers an unlocked task or contest when the player visits its starting zone. If the player accepts the offer, the objective is activated.
- Activated: The player can perform the stages of the task. Multiple objectives can be activated at once with different levels of progress in each.
- Tracked: The goals of the objective's current stage are displayed in the upper right corner of the screen. Only one objective can be tracked at a time, and the player can get credit for completing a goal only when the objective is being tracked.
- Completed: The player has completed all stages of the task and received the reward.

Add an Objective

To add an objective, start the Zone Settings Editor and open the `objectiveSettings` category. Add a new objective by clicking the green `+` to the right of `tasksSettings`, `contractsSettings`, or `contestsSettings`.

The resulting dialog box looks similar to the one used for `ZoneProperties`, but is used very differently. In the top field, enter the name for the objective.



Bug: The dialog box is called “Select Type”, but you actually use it to name your objective.

The objective name is used for the game to internally track the status of the objective, and it is also the name shown to the player. After the objective is added, you can edit its name by double-clicking it.

Tip: Localization potentially allows you to show the player the same name for objectives with different internal names, although this could be slightly confusing to the player.

The bottom field of the dialog box has a dropdown menu, but it only ever has one entry which matches the type of objective you are adding.

UI Text

I use the term “UI text” to refer to any text that the game shows to the player (such as the objective name).

UI text can include almost any characters, including Unicode characters. (For the truly esoteric characters, you’ll need to check whether they are supported by the game’s font.)

Special whitespace characters such as a tab or linefeed won’t display properly, but normal spaces are fine.

Although the double quote `"` and backslash `\` have special meaning in the `objective_settings.json` file, you don’t need to worry about them when using the Zone Settings Editor. The Zone Settings Editor escapes it appropriately.

SnowRunner uses the square brackets [and] and the vertical bar | for its own purposes, so they should not be used in UI text. Using them risks screwing up the message displayed to the player.

SnowRunner also allows certain HTML-style tags to format some types of UI text. This book explicitly states which types of UI text support formatting tags. If nothing is stated, then tags are implicitly not supported. (E.g. the objective name above does not support tags.)

The portion of text to format is surrounded by an opening tag and a closing tag. E.g. if a goal description is `Deliver to the <y>Swamp</>`, then it is formatted as below:



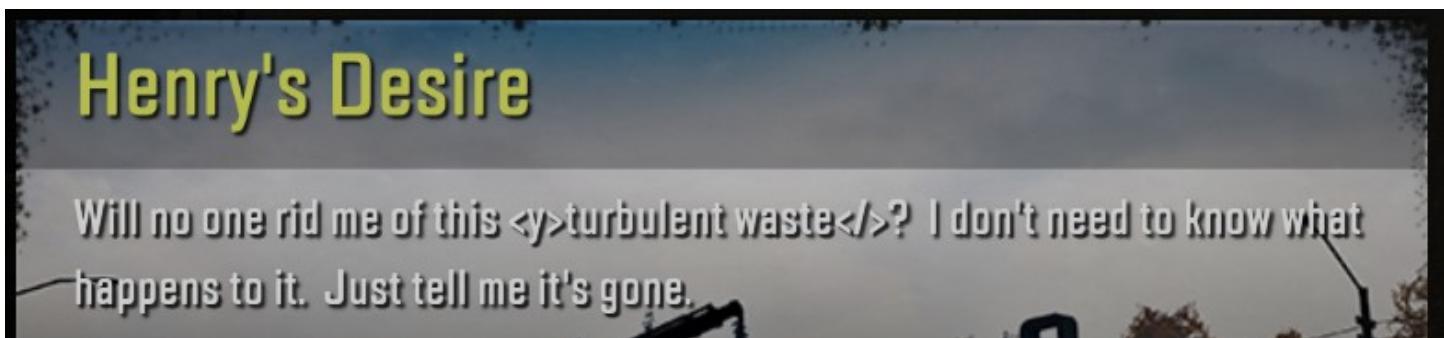
The formatting tags are as follows:

- `<y>` starts yellow text, and `</y>` ends it.
- `<r>` starts red text, and `</r>` ends it.
- `<g>` starts gray text, and `</g>` ends it.
- `<i>` starts italic text, and `</i>` ends it.
- `<u>` starts underlined text, and `</u>` ends it.

Tags can be nested, e.g. to get yellow italic underlined text. In that case, tags must be closed in the reverse order of how they were opened. E.g. `<y><i><u>SHOUTING</u></i></y>`. Unrecognized tags are quietly suppressed. I don't know of a way to escape < so that it appears in the text.

Bug: If the UI text ends after a < without a following >, the game will crash when it tries to display the text. This may happen even when formatting tags are not supported in that text type.

This book explicitly states which types of UI text support these formatting tags. If nothing is stated, then tags are implicitly not supported. E.g. in the objective description, tags have no effect on the offer text, but they do properly format the text in the objective details, **but** the raw tags are displayed in the completed task in the player's profile. So I consider tags to be unsupported in the objective description.



Saber has not yet documented the formatting tags, and the only tag that Saber uses in the main campaign is `<y>`. I would guess that all formatting tags are safe to use where supported, but you should pick your own risk tolerance.

Any UI text field can be localized into different languages as described on page 148.

Locked Objectives

The `Required rank` property specifies the minimum rank required to unlock the objective. The default `Required rank` is 0, which is less than the minimum rank that the player can possibly have, so the objective is unlocked as soon as the player starts the map. (A `Required rank` of 1 works equally well.)

An objective can also be locked because it is blocked by another objective or because it is implicitly blocked until unblocked. Details TBD.

Initial Contracts

TBD.

Offer Zone

A task or contest requires an offer zone. When the task or contest is unlocked and the player enters this zone, the game offers the objective to the player.

The primary offer zone is specified in the `taskGiverZone` property. This property requires a global zone ID, described in the next section.

Additional zones can be specified by adding properties under the `additionalTaskGivers` category. These properties require a regular zone ID.

When a task or contest is unlocked, its offer zone appears with a yellow perimeter. If the offer zone is shared with a zone type that normally has a red perimeter, the perimeter turns yellow until the objective is completed.

Bug: If two objectives are simultaneously offered in the same zone, the game only offers one to the player. The player cannot activate the second until she completes the first.

Bug: If the player's truck straddles the offer zones of two objectives (the same objective in both zones or two different objectives), the game gets confused and may offer neither objective until the player leaves both zones and then returns to just one of them.

Tip: To avoid triggering the above bugs, make sure that objectives are offered sufficiently separate in space or use blocking to offer them at separate times.

Tip: It is also helpful to offer objectives separate from other interactive zones so that the user doesn't need to switch among interaction types. This separation doesn't need to be as far, however.

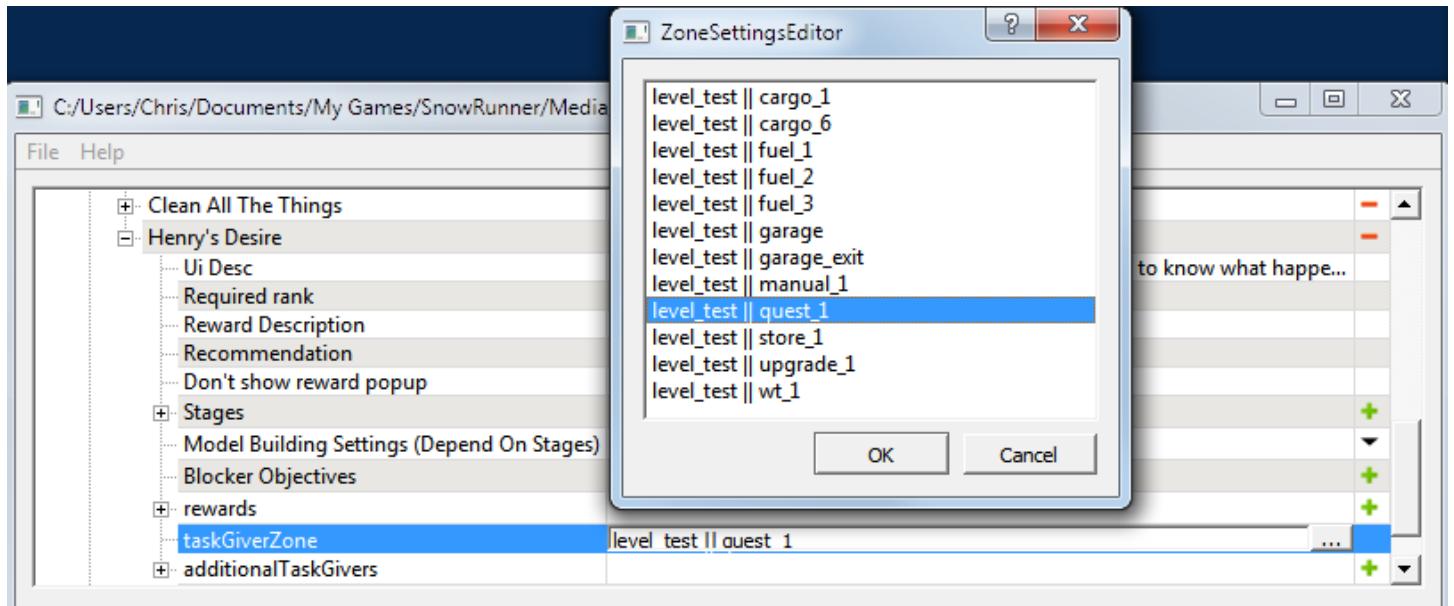
A contract doesn't have an offer zone because an unlocked contract is automatically activated.

Global or Regular Zone ID

Objectives can cross maps within a region. As such, certain properties require a global zone ID in which both the map name and the zone ID are specified. The map name is separated from the following zone ID by two vertical pipe characters, e.g. <code><level_name> || <zone_id>.

If the map is implicit, only a regular zone ID is needed. In this case, the map name should **not** be attached to the zone ID.

For either case, when you select the zone ID value for editing, a small button labeled appears to the right. Click the to open a dialog listing the known global or regular zone IDs, as appropriate. From this dialog, you can simply select the desired zone without worrying about its format.



Objective Overview

When the player views the objective offer, the game shows an overview of the objective and its rewards.



The `Ui Desc` property specifies the objective's description that is shown to the player. UI text is described on page 125. If the UI Desc value is blank, nothing is displayed in that area.

Rewards are described on page 143.

If the player views the objective details for an activated objective (or an offer), the game shows the above information, the goals for the current stage (or the first stage for an offer), and any recommended equipment for the objective.



Objective stages are described in the next section.

The **Recommendation** property allows you to choose a recommended piece of equipment for the player to have. No more than one recommendation can be selected.

Stages

An objective is split across one or more stages which the player must complete sequentially. Add a stage by clicking the green **+** to the right of the **Stages** container.

Each stage is comprised of one or more goals which the player may complete in any order. The goals in a stage can all be the same type or a mix of types.

For each added stage, all of the available goal types are listed. The method of adding each type of goal and the details for each type are described below.

In addition to having goals, a stage can also spawn cargo as described on page 137.

Cargo Delivery Goal

A stage can require the player to deliver certain cargo to one or more zones.

To create a cargo delivery goal, click the black arrow to the right of `actions`. This pops up a useless dialog.

Click `OK` to confirm it.



The `actions` category now has a `zoneToFill` subcategory. Click the green `+` to the right of `zoneToFill` to add one or more a cargo delivery goals.

Stages	
[-] [0]	[+]
[-] actions	ActionPackSettings
[-] zoneToFill	[+]
[-] [0]	[+]
Ui Desc	Deliver to the <y>Swamp</y>:
Truck need to visit Uid:	
globalZoneId	level_test store_1
Manual Unloading Settings	null
[-] cargo	CargoPack
name	CargoRadioactive
Amount	1
Model Building Settings (Depend On Cargo)	null
additionalUnloadZones	[+]
truckDelivery	[+]
visitAllZones	null
changeTruck	null
repairTruck	[+]
Seismograph Settings	null
spawnCargoOnStageActive	[+]
livingAreaInfo	null
Model Building Settings (Depend On Stages)	null
Blocker Objectives	[+]

Goal Description

The `UI Desc` property describes the goal for the player. It supports formatting tags. UI text and formatting tags are described on page 125. If the `Ui Desc` value is empty, the game displays an error instead.



Tip: The main campaign uses the `<y>` tag to highlight named locations on the map in the manner shown above, and you may wish to do the same. Notice how the `:` sets up the cargo to deliver, which the game always writes immediately after the goal description.



Cargo to Deliver

To specify the cargo to deliver, click the black arrow to the right of `cargo` and confirm the useless dialog. Specify the cargo type and amount in the subproperties that appear.

The `name` subproperty specifies a cargo type. When editing this property value, you can either type in a name or select a name from the dropdown menu. Note that the menu is missing some names, so you'll need to type those in by hand. Reference information for cargo types is on page 171.

The `Amount` subproperty specifies how much cargo is required to complete the delivery. The player is allowed to deliver less than the full amount at once, so she can make multiple trips to fulfill a large objective.

Bug: The properties don't support multiple cargo types within a single goal. Instead you must add more goals under `zoneToFill`, each with their own description, etc.

Delivery Zones

The `globalZoneId` property specifies the global zone ID for the primary delivery zone. If the player is allowed to deliver the cargo to other zones instead, these are specified with regular zone IDs in `additionalUnloadZones`.

Tip: `additionalUnloadZones` is useful for making a pair of delivery zones on either side of a blocked passage.

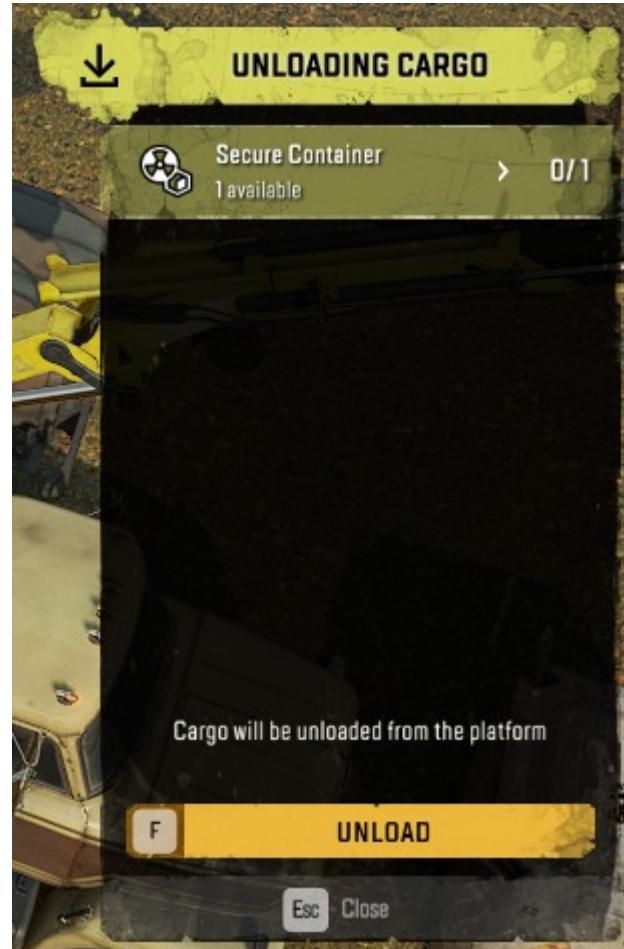
If `Manual Unloading Settings` is `null`, the player can only complete the delivery while the required cargo is packed on her truck or trailer.

Manual Unloading

To require manual unloading, click the black arrow to the right of `Manual Unloading Settings` and confirm the useless dialog. This adds a subproperty, `Interaction zoneUid:`. Fill this subproperty with a global zone ID. In order to complete the goal, the player's truck must be in this interaction zone (`Interaction zoneUid:`), **and** the cargo must be unpacked and within the primary delivery zone (`globalZoneId`).

Bug: When manual unloading is required, the zones in `additionalUnloadZones` are made visible as if they are additional delivery zones, but they offer the cargo management dialog as if they are additional interaction zones. However, these zones don't actually work for either function: cargo unloaded in them is not recognized as deliverable, and their cargo management dialog never recognizes when the cargo is correctly placed in the delivery zone. So keep `additionalUnloadZones` blank when using manual unloading.

The below screenshots show the cargo management dialog before and after the cargo is correctly unloaded. Your milage may vary, but I found the grammar of the first dialog extremely confusing until I noticed the tiny red period that ends the first sentence.



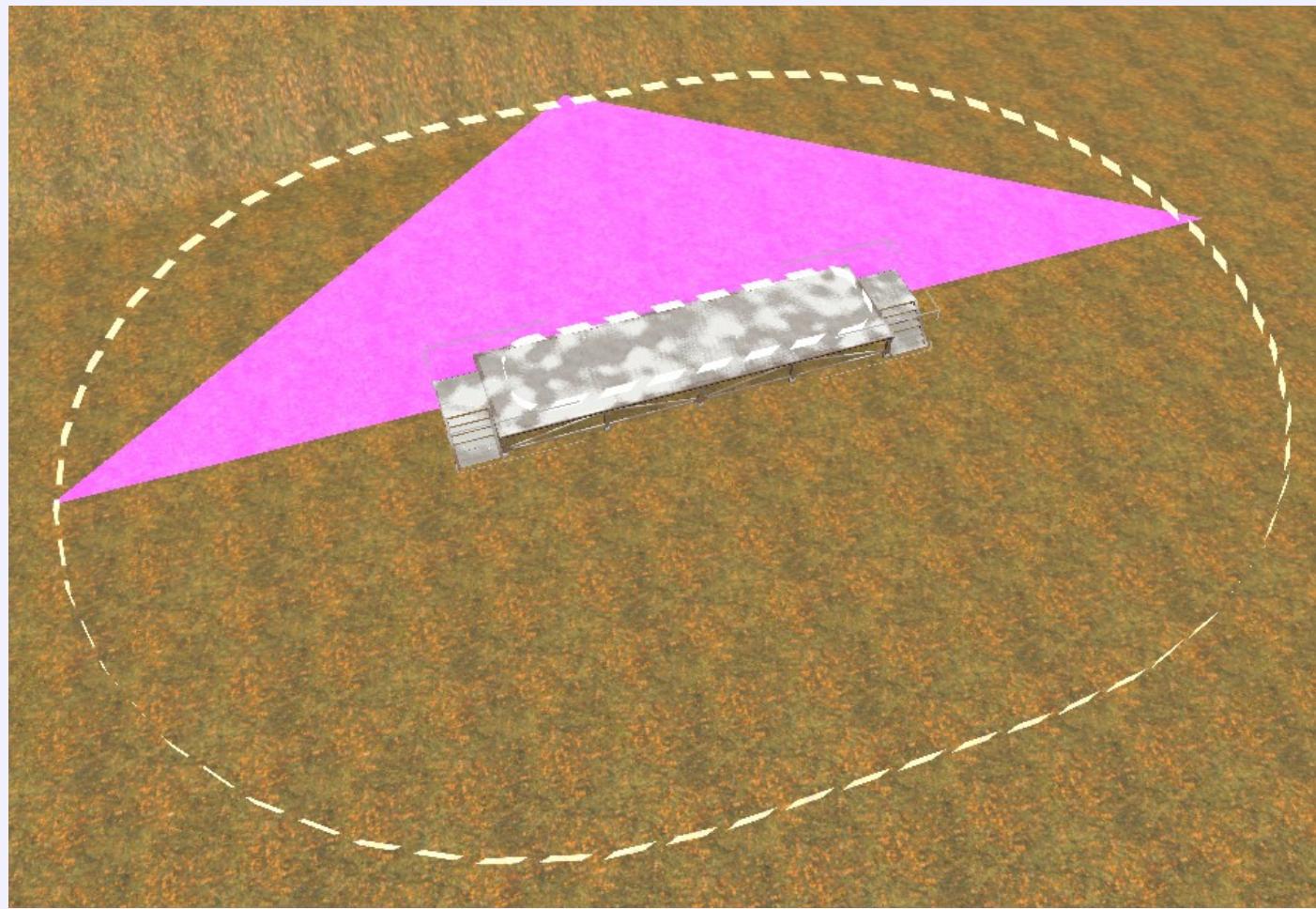
The delivery zone is made visible in the driving view while the objective is activated.

Warning: The interaction zone ([Interaction zone Uid:](#)) is not visible in the driving view (unless the zone has another purpose that makes it visible).

Saber recommends that the interaction zone entirely surrounds the delivery zone so that as soon as the player finishes unloading the cargo, she can make the delivery from the cargo management dialog offered by the invisible interaction zone.

Tip: The interaction zone should be large enough so that the delivery can happen even after a long stretch with a big crane, but small enough so that the delivery zone is obvious when the cargo management dialog is offered. I find that the longest crane stretch for both US and Russian is about 13.5 meters. Appropriate interaction zone dimensions can thus be calculated as follows:

- interaction zone `Length` = delivery zone `Length` + 27
- interaction zone `Width` = delivery zone `Width` + 27
- interaction zone `Rounding radius` = delivery zone `Rounding radius` + 13.5



If the player can drive into the delivery zone, she can potentially make the delivery by simply unpacking the cargo within her truck without actually unloading it.

Tip: Place the delivery zone on a raised platform (such as the various `platform` models) that the truck cannot climb onto.



The interaction zone and delivery zone are both shown on the navigation map when the objective is being tracked or when the objective details are viewed.



Tip: To avoid a pile-up of names and icons, set the interaction zone's [Is Visible on Minimap](#) property to [False](#).



Manual unloading goes well with cargo that is spawned for the stage and thus requires manual loading. See

Bug: The game treats a dynamic model of cargo as different than unpacked cargo. So if you place a cargo model on your map, then use a crane to transport it directly to a manual unloading zone, the game won't accept it. You have to load it onto your truck, pack it, then unpack it. This transforms it into the unpacked cargo that the game accepts. This means you probably shouldn't have an objective that requires the player to move loose cargo only a short distance.

This bug doesn't affect cargo spawned from a manual cargo loading zone, which spawns the cargo as unpacked cargo.

Required Truck

The `Truck need to visit Uid` property is optional. If filled, it should refer to a truck's unique ID (page 64), and the truck should not be reserved (page 66). The player is then allowed to make the cargo delivery only in this truck. Note that the ID refers to a specific truck, not a truck type. If a truck type is put in this field, then it won't match a truck ID, and so no truck can make the delivery.

The specified truck must not be reserved for another goal. Reserved trucks are described on page 66.

Bug: The game never tells the player which truck is required, or even that a specific truck is required at all. Instead, when the player arrives at the delivery zone with the cargo in the wrong truck, the game either doesn't offer the cargo management dialog, or it allows the player to select the cargo to deliver and then ignores their keypress to deliver it. You'll need to tell the player in the `UI Desc` which truck to use.

Bug: If the player ever visits a garage with that specific truck, its truck ID is lost. I.e. when the player leaves the garage, the game doesn't respawn the original truck, but instead spawns a new truck of the same type. So to use `Truck need to visit Uid` with any safety, you need to make sure that no garage is unlocked, or make sure that the truck is securely walled off from being able to visit a garage.

Spawn Cargo for Stage

Loose cargo can be spawned when a particular objective stage begins. This is the best mechanism for encouraging the player to manually load cargo for a cargo delivery goal in the same stage. Since the cargo is respawned if the stage is restarted, the player can never get stuck without sufficient cargo.

To spawn cargo when a stage is started, click the green **+** to the right of `spawnCargoOnStageActive`. You can add as many spawned cargo as you like.

<code>spawnCargoOnStageActive</code>	
<code>[0]</code>	+ -
<code>Don't show reward popup</code>	<input type="checkbox"/>
<code>zone</code>	<code>level_test manual_1</code>
<code>cargos</code>	+ -
<code>[0]</code>	
<code>name</code>	<code>CargoServiceSpareParts</code>
<code>Amount</code>	<code>1</code>
<code>Should be discovered by metalldetector</code>	<input type="checkbox"/>

Set the `zone` property to the global zone ID of the zone where the cargo should spawn. Cargo spawns in this zone much like in a manual loading zone (page 122). However, instead of asking for confirmation to delete other cargo in the zone, the cargo in this case fails to spawn.

Bug: The game does not indicate to the player that cargo failed to spawn, nor does it indicate **where** it failed to spawn, so the player has little hope of figuring out how to correct the situation. If the cargo spawns near where the stage starts (e.g. the task or contest offer zone), the player **might** figure out that the stray cargo she can see is obstructing the spawn of cargo she can't see, but then again she might not.

Interestingly, if a truck is in the spawn zone, it doesn't prevent cargo from spawning. If the cargo intersects a truck, it is displayed as a "ghost", and only becomes real when the truck is moved out of the way. The ghost cargo is shown on the map the same as if it is real.

Tip: Spawn cargo only in an area where it is highly unlikely that the player will drop loose cargo. Make the spawn zone's length and width very small or perhaps 0, although cargo can obstruct even a zero-size zone if the cargo is close enough.

Click the green **+** to the right of `cargos` to add exactly one entry to the category. Only one cargo item can be spawned in a zone, so there's no point in listing more. Set the `name` of the cargo type to spawn as usual. The `Amount` is hard-coded to 1 and cannot be edited.

The game works best when the spawned cargo types match the cargo delivery requirement for the same stage and when there is no other source for the same cargo type. In particular, nothing stops the player from ignoring the spawned cargo and instead delivering some other cargo of the same type.

If the checkbox next to `Should be discovered by metalldetector` is enabled, then the spawned cargo cannot be seen on the map until discovered by the metal detector. Details TBD.

no reward popup TBD.

Cargo Models

If you place a model on the map with one of the cargo model types, I call it a ‘cargo model’. This distinguishes it from cargo that is spawned from a manual loading zone or from an objective stage, which I call ‘unpacked cargo’.

Bug: A cargo model isn’t labeled on the navigation map, and it cannot be delivered. If it is lost, it cannot be respawned.

The player can convert a cargo model to regular cargo by lifting onto a vehicle with a crane and packing it. It can then be delivered to a destination that doesn’t require cargo unloading, or it can be unpacked for manual unloading. However, if the player attempts to transport it directly to the manual delivery zone without ever packing the cargo model, she won’t be able to deliver it.

Cargo models don’t disappear after a collision like other dynamic models (page TBD), but the player can remove a cargo model in the same way as other packed or unpacked cargo. But like any other dynamic model, a cargo model stays in its original position on the navigation map regardless of how it is moved or transformed.

All of these factors make cargo models poorly suited for cargo delivery goals. If you choose to place a cargo model on your map (e.g. as scenery), consider making it static to eliminate any potential confusion for the player.

Cargo Sources on Map

Cargo and cargo loading zones are highlighted in various ways on the navigation map and in the objective details, depending on the tracked objective. Understanding this highlighting can help you decide how simple or complex you want to make the cargo situation on your map.

However, before we can talk about highlighting, let’s review the normal visibility of cargo on the map:

- ‘Recognized’ cargo has a 3-D shape on the map with an icon and label. If unpacked cargo is in a revealed area of the map, it is recognized.



- ‘Unrecognized’ cargo has a 3-D shape on the map, but has no icon or label. Unpacked cargo that is in the gray, unrevealed part of the map is unrecognized. A cargo model (described in the section above) is also unrecognized.



- ‘Invisible’ cargo cannot be seen on the map at all. Cargo that is in a cloaked part of the map is invisible. Cargo that is packed on a truck or trailer is also invisible.

A cargo loading zone never has a 3-D shape on the map, but it can be ‘recognized’ with an icon and label if it is revealed and `Is Visible on Minimap` is not `False`. Otherwise it is invisible.

OK, now we can discuss highlighting of cargo and cargo loading zones (cargo sources):

- A ‘highlighted’ cargo source is marked with a yellow ‘load’ icon on the navigation map when its objective is tracked or when the objective details are viewed.



- A ‘recommended’ cargo source is always highlighted as above. In addition, when the objective details are viewed, each recommended cargo source is listed under the delivery goal description to the left of the map. Selecting a recommended cargo source centers the navigation map on that source.



- An ‘unhighlighted’ cargo source is not marked with a yellow ‘load’ icon on the navigation map. An unhighlighted cargo source is never recommended.

Cargo Sources when cargo was spawned

If cargo was spawned by the current stage of the tracked objective, then cargo is recommended based on the cargo types required by the current delivery goals, regardless of whether or not they match the cargo types that were spawned.

If the player views details for the objective while it is being offered and before it is activated, then cargo has not yet spawned, so cargo is highlighted as per page 142.

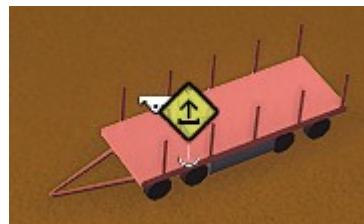
Any unpacked cargo that matches one of the required delivery types is recommended. If there is more than one recommended cargo source, then each is listed below the delivery goal description in the objective details to the left of the navigation map. This may be a different number of cargo items than is listed under the tracked objective in the upper right of the screen.

Note that a cargo source can be recommended even if it is invisible.



If a cargo type is required for delivery, but there is no unpacked cargo of that type to recommend, then the cargo type is listed in the objective details, but selecting it does not center the map on anything.

Besides the recommended cargo sources above, other cargo sources are also highlighted. A cargo loading zone is highlighted if it includes a cargo type used in the current delivery goal and `Is Visible on Minimap` is not `False` for the zone. The cargo loading zone remains highlighted even if it has run out of inventory of the needed cargo type. Cargo packed on a detached trailer is also highlighted.



Bug: Unpacked cargo that was spawned by the current objective stage is highlighted, as is other unpacked cargo of the same type, even if it is not required by a delivery goal. It may also disrupt highlighting of unpacked cargo that **is** required, although at this point it gets so confusing that I'm not sure anymore.

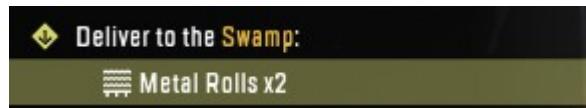
That leaves cargo that isn't highlighted:

- Unpacked cargo that isn't a type required by a delivery goal and doesn't match the type of spawned cargo.
- Cargo packed on a truck or on an attached trailer.
- Any cargo model.

Tip: The recommended cargo sources make the most sense when exactly the right amount of cargo is spawned as is required for delivery, and there are no other sources for that cargo type.

Cargo Sources when cargo was not spawned

If no cargo was spawned by the current stage of the tracked objective, then only cargo loading zones and detached trailers with packed cargo are highlighted. The cargo details lists the quantity of each cargo type required for delivery on a single line. Selecting that line centers the map on a random highlighted cargo loading zone. The player has to zoom out and pan the map around to find any other highlighted cargo loading zones or detached trailers.



Truck Delivery Goal

A stage can require the player to deliver one or more particular trucks to specified zones.

To create a cargo delivery goal, click the green **+** to the right of `truckDelivery`. This adds one truck delivery goal.

<code>truckDelivery</code>		+
<code>[0]</code>		-
<code>Truck Uid</code>	<code>clean_truck_1</code>	
<code>Truck UI Name</code>	<code>The Dirty Truck</code>	
<code>afterStageFinished</code>	<code>REWARD</code>	
<code>Ui Desc</code>	<code>Get that truck over here.</code>	
<code>globalZoneDeliveryId</code>	<code>level_test quest_1</code>	
<code>additionalDeliveryZones</code>		+
<code>[0]</code>	<code>fuel_3</code>	-

The `truck Uid` should specify a truck's unique ID (page 64). When a truck's ID is used in a truck delivery goal, that truck is reserved (page 66), and the player cannot drive it until some objective awards it to the player. Since a player cannot drive the specified truck, the player can only complete the truck delivery goal by towing it with another.

The `Truck UI Name` provides a name for truck to use on the navigation map and in the objective details. Formatting tags TBD.

figure TBD

The `UI Desc` property describes the goal for the player. It supports formatting tags. UI text and formatting tags are described on page 125.

figure TBD

The `globalZoneDeliveryId` property specifies the global zone ID for the primary delivery zone. If the player is allowed to deliver the truck to other zones instead, these are specified with regular zone IDs in `additionalDeliveryZones`.

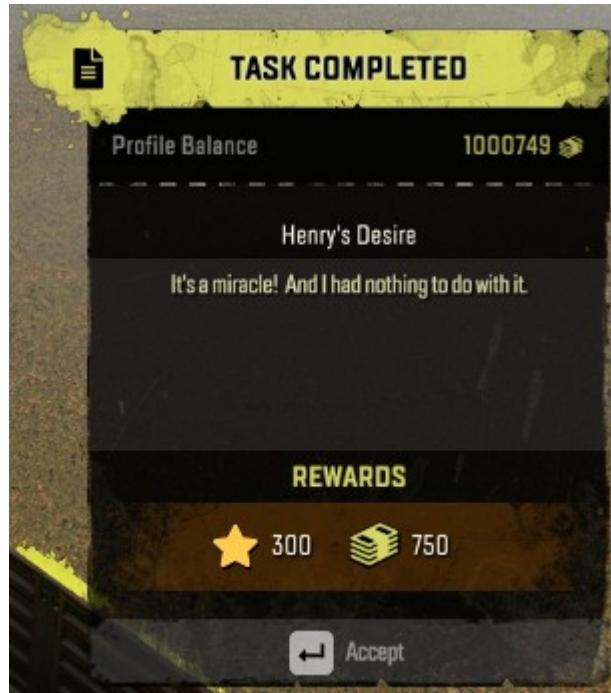
Tip: `additionalDeliveryZones` is useful for making a pair of delivery zones on either side of a blocked passage.

Truck Repair Goal

Rewards

Completion

The `Don't show reward popup` property has a checkbox that can be enabled or disabled. When enabled, music plays when the player completes the objective, but no information appears onscreen. When disabled, the game pops up a message when the objective is complete. The rewards are listed, along with the text in the `Reward Description` property.



Once the objective is completed, it is removed from the navigation map. Any zones that were only being used for that objective are hidden.

you don't specify a value for `Ui Desc`, the game will display an error instead.



Models

Individual Plants

Distributions

Localization

The various name and description properties of your map features can be localized into different languages. By default, each property value is simply the text to display. However, the property value can instead be an ID that is cross-referenced into a file that is different for each supported language. For example, the name FREDs_GARAGE can display as “Fred’s Garage” for someone playing in English or “Гараж Фреда” for someone playing in Russian.

To support localization, a file for each desired language must be placed in the `Media/prebuild/<level_name>/texts` directory. If you don’t have this directory yet, you’ll have to create it. The filenames for each language are as follows:

```
strings_brazilian_portuguese.str  
strings_chinese_simplified.str  
strings_chinese_traditional.str  
strings_czech.str  
strings_english.str  
strings_french.str  
strings_german.str  
strings_italian.str  
strings_japanese.str  
strings_korean.str  
strings_polish.str  
strings_russian.str  
strings_spanish.str
```

These files must use the little-endian UTF-16 encoding. The easiest way to ensure this encoding is to copy and modify an existing file. Windows Notepad or Emacs can read little-endian UTF-16 and will automatically write the file back out in the same format. To get an example file to start with, choose `File→Create default truck mod` in the Editor, then copy one of the files from `Media/Mods/<truck_name>/texts`.

Each line of a localization file is an ID followed by the localized text enclosed in double-quotes, e.g.

FREDS_GARAGE	"Fred's Garage"
--------------	-----------------

For safety, I recommend that the localization ID be limited to a combination of lowercase `a – z`, uppercase `A – Z`, `0 – 9`, and `_`. It should not include other characters, including spaces.

The localized text can be almost any Unicode text, but with the following restrictions. If you want to use a standard a double-quote `"` in the quoted localized text, escape it with a backslash, i.e. `\"`. A backslash should also be escaped with another backslash, i.e. `\\"`.

As with non-localized text, special whitespace characters such as a tab or linefeed won't display properly, but normal spaces are fine. SnowRunner uses the square brackets [and] and the vertical bar | for its own purposes, so they should not be used here, even with a backslash escape. If this book says that formatting tags are supported in the text, then formatting tags are equally supported in the localization. The left angle bracket < should only be used as part of a supported formatting tag.

Bug: If the UI text ends after a < without a following >, the game will crash when it tries to display the text. This may happen even when formatting tags are not supported in that text type.

The separator between the ID and the localized text can be any mix of spaces and tabs, although Saber recommends tabs only.

Note that if an ID isn't found in the appropriate localization file, the ID itself is displayed. Thus, you can always simply specify the desired text directly in the property value rather than using a localized ID.

Bug: If SnowRunner can't find the desired localization for a particular ID, it displays the ID itself even if a localization exists in English or another language. Thus, if you've localized your map into only a subset of languages, the best option for the other languages may be to simply copy over the English file.

Playing Your Map

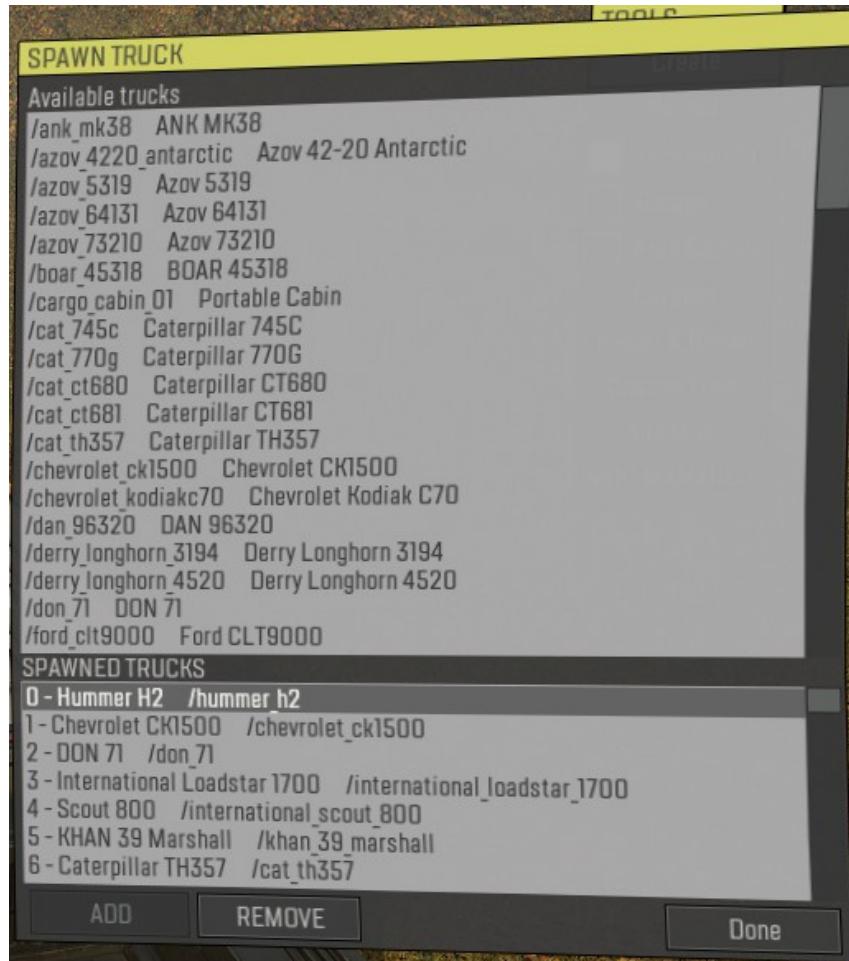
Dev Tools

While running the game in a proving grounds map or a local custom map, a **Tools** menu is shown. This menu of development tools (abbreviated as “Dev Tools”) gives you the ability to quickly try different combinations of trucks, trailers, and add-ons.

The various options in the **Tools** menu are described below.

Dev Tools: Create

Clicking **Create** pops up a dialog that allows you to create a truck and/or warp to any existing truck.



To add a truck, click on any truck under **Available trucks**, then click somewhere on the visible terrain. A ghost image of the selected truck will appear. Rotate the truck with left click and drag. When you are satisfied with its location, click **ADD**.

Warning: the created truck may intersect hazards on the map such as models or other trucks. As soon as you get close enough to enable its physics, the truck may take (or cause) significant damage.

To remove the truck currently being driven, click **REMOVE**.

Warning: if you close the **Create** dialog while not in control of a truck, you won't be able to drive any truck or even move the camera. However, you can still use the **Tools** menu to jump into another truck.

To jump into another truck from the **Create** dialog, click any truck listed under **SPAWNED TRUCKS**.

Bug: While stranded outside of a truck, the map won't list any trucks, so you can't jump to a truck that way.

Dev Tools: Reload

The **Reload** option is not particularly useful to map development. It assists truck modders by reloading truck assets without leaving the map. It is not possible to change a packed map while it is being used by the game.

Dev Tools: Information

The **Information** checkbox puts some velocity information in the center of the HUD. Again, this is useful only to truck modders.

Dev Tools: Virtual Garage

The **Garage** checkbox pops up additional menus that allow you to customize the current truck with add-ons and trailers. This virtual garage allows any compatible part to be installed regardless of whether you unlocked the part or reached the necessary rank.

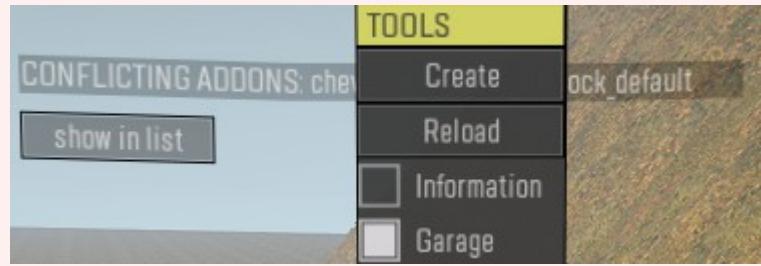
Bug: You can open the map while in the virtual garage, but it will not give you the usual options to view objectives or change trucks. Close **Garage** mode and re-open the map to get the usual map options.

All parts in the virtual garage are listed by the lowercase name used by the Editor. Thus, the **Garage** option is extremely useful for getting the names of compatible parts to add to trucks in the Editor.

To install or uninstall an item, double-click it.

If there is an installation conflict, a message will appear on the screen along with a button, `show in list`. Clicking `show in list` moves the part selection to the conflicting part (or one of the conflicting parts if there are multiple). The conflicting part may need to be uninstalled (to make room) or installed (to provide an attachment point, such as for a semi-trailer). You can then double-click the conflicting part to install/uninstall it before re-attempting your original change.

Bug: The conflict message is obscured by the `Tools` menu. Therefore, you'll probably need to click the `show in list` button to find the (first) conflicting item.



If your truck is too close to another object, the installation may instead fail because there isn't enough room to install it without causing a collision. In this case, the part will not be installed. This can be distinguished from a conflict with another part by the lack of a conflict message.

Tires, Rims, and Suspension

Tires and rims must match. To make this easier, installing any tire will automatically install a compatible rim if necessary. This helping hand only goes one way, though. If you attempt to install a rim that is incompatible with the current tire, the install silently fails.

The tire and rim size must also be compatible with the suspension. If you attempt to install a tire size that is incompatible with the suspension, the install silently fails.

Bug: After installing a raised suspension and a large tire and rim, the virtual garage will allow you to install a shorter suspension while keeping the oversized tire and rim. Do not attempt to use this incompatible combination in the map Editor.

Dev Tools: Free Camera

Clicking the `Free Camera` checkbox separates the camera from the truck. There is no immediate change to the camera view, but you can now drive away from the camera. Or you can move the camera freely using the following controls:

- right click and drag: rotate the camera up, down, left, or right.

- **I** or **NumPad 8**: move the camera forward.
- **K** or **NumPad 2**: move the camera backward.
- **J** or **NumPad 4**: move the camera left.
- **L** or **NumPad 6**: move the camera right.
- **O**, **NumPad 3**, or **Shift**: move the camera up.
- **U**, **NumPad 1**, or **Z**: move the camera down.

Dev Tools: Cargoes

Click **Cargoes** in the **Tools** menu adds some information to the upper left showing the usage of cargo points on the truck.

Unfortunately, there is no way to artificially spawn cargo using the **Tools** menu.

Dev Tools: Repair & Refuel

Click **Repair & Refuel** to fully repair and refuel your current truck. The parts and fuel are not taken from any truck; they are gifted to you by the **Tools** menu.



Dev Tools: Change Time

Click **Change Time** to cycle through the lighting conditions at different times of day. This is useful for checking how your map looks when streetlights are on or off, at dusk, etc.

Dev Tools: Mod Tools

The **Add Mod** and **MOD MANAGER** buttons allow modded trucks to be managed and spawned.

Revealing the Map

The map is initially an unexplored land. As the player drives around and visits watchpoints, more details are revealed in the map view, and vehicles and zones are discovered. By default, the map view is accessed by pressing the **M** key in the game.

To fully reveal an area, the player must drive a truck near it. Areas within 39 meters of the active truck are drawn in color in the map view. The player can also visit a watchpoint, which reveals the map within the radius

set by the watchpoint's property. Vehicles and zones are labeled on the map if they are within a fully revealed area.

If the map is set to be uncloaked, then the portions of the map that have not been fully revealed are partially revealed. These areas of the map are drawn in gray shades. The shapes of vehicles can be seen in these partially revealed areas, but they are not labeled on the map. Zones are also not labeled on the map if they are only partially revealed.

If the map is set to be cloaked, then most of the map is initially black. As the player drives around, areas of the map within about 140 meters of the active truck are partially revealed. Visiting a watchtower will fully reveal some portion of the map, but it does not partially reveal any additional area.

Discovery of Vehicles

If a truck or trailer is set to be locked, then even after it is revealed, its label is initially gray in the list of **TRUCKS & TRAILERS** on the left side of the map.



A player can “discover” the vehicle by driving within 15 meters of it (as measured between the truck centers). This discovery rewards the player some experience points, and its label changes to white on the left side of the map. If a locked truck has not yet been discovered, the player can't jump into it, even if the target truck is clearly visible. Once the player discovers the truck, she can jump into it by using either the map view or the **CHANGE TRUCK** function.



If a truck or trailer is set to be unlocked, then its label is white on the left side of the map as soon as it is revealed. There is no separate “discovery”, and nothing new happens when the player drives near it. As soon as an unlocked truck is revealed, the player can jump into it from the map view. Alternatively, the player can use the **CHANGE TRUCK** function to jump into an unlocked truck. In this case, the target truck does not necessarily need to be revealed as long as it is currently with about 50 meters from the camera. (I.e. moving the camera affects whether you can jump into a truck.)

Appendix A: Reference Information

The following sections contain various lists and tables that are handy for reference but would have cluttered up the main text. Once you're familiar with the editor, you may find yourself using this appendix the most often.

Truck Reference

The table below provides overview information for all trucks so that you don't have to try them all individually. To get detailed information about a particular truck, use the dev tools to create the truck and view it in the virtual garage to discover all allowed add-ons for the truck.

Another source of vehicle information is the [SpinTires fandom wiki](#).

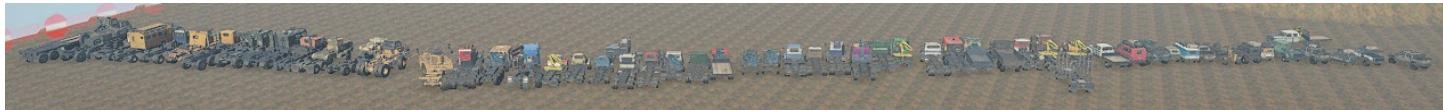


Table Column Meanings

Truck is the name of the truck in the editor and the dev tools. The name of the truck in the game is similar. (In fact, the truck names are virtually identical across all languages, even Russian.)

Reg. indicates the region that the truck is associated with.

- US is the United States/North America.
- Rus. is the Russian Federation.

AWD indicates the truck's all-wheel-drive capability.

- no: AWD cannot be enabled
- upgrade: AWD can be installed after the appropriate upgrade is obtained
- yes: AWD can be enabled by the player
- always: AWD is always enabled

Diff Lock indicates the truck's diff-lock capability.

- no: diff lock cannot be enabled
- upgrade: a locking differential can be installed after the appropriate upgrade is obtained
- yes: diff lock can be enabled without an upgrade, but only in a low gear
- yes+no: yes by default, but can be removed with an "upgrade"
- always: the differential is always locked

Tires indicate what types of tires the truck can use.

- H: highway
- A: all-terrain
- O: offroad
- M: mud
- C: chains

Trailer indicates what kinds of trailers the truck can pull.

- S: scout trailers
- T: regular trailers
- L: low-saddle semi-trailers
- H: high-saddle semi-trailers
- R: special rock semi-trailer

Bed indicates what kind of non-log cargo the truck can carry in its bed (or on the roof).

- B: big crane
- R: repair parts; if followed by a *, can be combined with a saddle
- r: a small number of repair parts; sometimes can be combined other bed items. Spare wheels are not covered in this reference.
- F: fuel tank
- f: a small amount of fuel; sometimes can be combined other bed items
- M: maintenance parts (repair + fuel)
- m: a small amount of maintenance parts (repair + fuel); sometimes can be combined other bed items
- D: metal detector
- V: seismic vibrator
- C: mini-crane. If followed by a '+', can be combined with any following options (usually only cargo units). Always conflicts with semi-trailer saddles.
- 1–4: 1 to 4 generic cargo units
- G: can carry a cargo container, but not other cargo units

Logs indicates what kind of logs the truck can transport.

- C+: log crane; can be combined with S, M, or L if available
- C/: log crane; can be combined with M if available, but not S or L
- S: short logs in the bed; can be combined with medium logs if a trailer is available.
- M: medium logs in a trailer
- L: long logs stretching from the bed to a trailer

DLC indicates the DLC that is required to use the truck.

- S1: Season 1: Search & Recover
- S2: Season 2: Explore & Expand
- S3: Season 3: Locate & Deliver
- S4: Season 4: New Frontiers
- AD is the Anniversary DLC (free)
- TD* is the Tatra Dual Pack
- CL* is the Classico Pack
- GB* is the GMC Brigadier DLC
- N5* is the Navistar 5000-MV DLC (limited availability)
- WS* is the Western Star 49X DLC
- * indicates that I don't own these DLCs, so my data might not be perfect.

Some truck add-ons were also released as part of a DLC package, but you don't need to own the DLC to get the add-ons. DLC is required only for the truck itself.

Scout Trucks

Reg.	Truck	AWD	Diff Lock	Tires	Trailer	Bed	Logs	DLC
US	cat_th357	always	always	M		C		S2
	chevrolet_ck1500	yes	upgrade	HAOMC	S	rfm		
	chevy_apache	yes	always	AOMC		m		CL*
	ford_f750	upgrade	always	HOC	T	C+1	M	S1
	hummer_h2	yes	upgrade	HAOMC	S	mm		
	international_loadstar_1700	always	always	HAOC	S	CRm		
	international_scout_800	yes	always	HAOMC	S	m		
Rus.	don_71	yes	yes	HAOMC	S	fm		
	khan_317_sentinel	yes	yes	HAOMC	S	f		S4
	khan_39_marshall	always	yes	M		m		
	khan_lo4f	always	always	HAOMC	S	m		
	tuz_166	always	always	HAOMC	S			
	tuz_420_tatarin	always	always	O				
	yar_87	yes	always	AOMC	S	r		

Highway Trucks

Reg.	Truck	AWD	Diff Lock	Tires	Trailer	Bed	Logs	DLC
US	ford_clt9000	no	no	HAOC	TLH	BRFMDC+2	C/SML	
	gmc_9500	upgrade	upgrade	HAOC	TLH	BRFMDVC+2	C/SML	
	international_transtar_4070a	no	no	HAOC	TLH	BRFDC+2	C/SML	

Heavy Duty Trucks

Reg.	Truck	AWD	Diff Lock	Tires	Trailer	Bed	Logs	DLC
US	cat_ct680	upgrade	yes+no	HAOC	TLH	BRFMDC+2	C/SML	AD
	cat_ct681	upgrade	upgrade	HAOC	TLH	BRFMC+2	C/SML	
	chevrolet_kodiakc70	upgrade	yes	HAOC	TLH	BRFDC+2	C/SML	
	gmc_8000	upgrade	no	HAOC	TLH	BRFMDVC+2	C/SML	GB*
	international_fleetstar_f2070a	upgrade	yes	HAOC	TLH	BRFMDC+2	C/SML	
	international_hx_520	upgrade	no	HAOC	TLH	BRFMC+2	C/SML	AD
	ws_4964_white	upgrade	upgrade	HAOC	TLH	BRFMC+2	C/SML	

Offroad Trucks

Reg.	Truck	AWD	Diff Lock	Tires	Trailer	Bed	Logs	DLC
US	ank_mk38	always	always	HAOMC	T	2	M	
	freightliner_114sd	upgrade	yes	HAOC	TLH	BRFMDVC+2	C/SML	
	freightliner_m916a1	yes	always	HAOC	TLH	C	M	
	international_paystar_5070	yes	yes	HAOMC	TLH	BRFMDVC+2	C/SML	
	royal_bm17	yes	no	HAOC	TLH	BRFMDVC+2	C/SML	
Rus.	azov_5319	always	always	AOMC	TLH	BRFMVC+2	C/SML	
	azov_64131	always	always	AOMC	TLH	BRFMVC+2	C/ML	
	krs_58_bandit	always	always	HAOMC	TLH	FMVC+Rr2	C/SML	S2
	step_310e	yes	yes	HAOC	TLH	BRFMDC+2	C/SML	
	tatra_805	yes	upgrade	HAOMC	T	m2	M	TD
	tayga_6436	always	always	HAOMC	TLH	BRFVC+2	SML	
	tuz_16_actaeon	yes	yes	HAOMC	T	RFC+1	ML	S1
	tuz_108_warthog	yes	yes	HAOMC	TL	RFC1	M	
	voron_ae4380	always	always	HAOMC	TLH	BRFDC+2	SML	
	voron_d53233	always	always	HAOMC	TLH	BRFMDC+2	C/SML	
	zikz_5368	yes	upgrade	HAOMC	TL	RFMD2C+1	C/ML	

Heavy Trucks

Reg.	Truck	AWD	Diff Lock	Tires	Trailer	Bed	Logs	DLC
US	cat_745c	yes	yes	M		FG	SM	
	cat_770g	no	always	M		F	R	S2
	derry_longhorn_3194	always	always	HAOC	TLH		M	
	derry_longhorn_4520	yes	yes	HAOC	TLH		ML	
	navistar_5000mv	always	yes	HAOC	TLH	C	ML	N5*
	pacific_p12w	yes	always	AOMC	TH	BRMD	C/SML	
	pacific_p16	no	always	O	TH		ML	
	pacific_p512	no	upgrade	O	TLH	BFMDVC+R2	C/SML	S3
	paystar_5600ts	upgrade	upgrade	HAOMC	TL	BR*FC+3	C+SML	
	western_star_49x	upgrade	yes	HAOC	TLH	RFMVC+2	ML	WS*
	ws_6900xd_twin	upgrade	yes	HAOMC		4		
Rus.	azov_4220_antarctic	always	yes	M	T	2	C/ML	
	azov_73210	always	always	AOMC	TLH	BRFMV3C+2	C/SML	
	boar_45318	yes	upgrade	HAOC	TH	RV	ML	S3
	dan_96320	always	always	AOMC	TLH	BRF2	ML	
	kolob_74760	always	always	M	TH		M	
	kolob_74941	yes	yes	M	TH		M	
	tatra_t813	yes	yes	AOMC	TH	m2	M	TD*
	zikz_605r	always	always	M	TH	BRrFV2	C/SM	S4

Bug: The cat_745c_log_bunk add-on can hold either short or medium logs, so for example a CAT 745 can be initialized with cargo_logs_cat_745c_log_bunk (short logs). However, due to a bug, the same short logs cannot be loaded onto the CAT 745 in the game.

Texture Layer Reference

Below is reference information for the various texture layers available in the Editor. Since the texture layer dialog box shows a preview of the ground texture, I won't cover that. But many texture layers share the same ground texture while having different 3-D models and/or different properties. I've highlighted the important differences below.

The recommended range for the **Tiling scale** property is derived from the values used in Saber's reference maps. A few maps use values outside of the ranges I've listed; feel free to experiment and use a value that looks natural in your setting.

Some layers have weird suffixes like _test and _out that look like they were released by mistake. But most of these are used extensively in Saber's reference maps, so it's more likely that these materials simply got used in too many places and so weren't worth the trouble of renaming. I think you can feel comfortable using them in your own maps.

TBD: review special properties in XML files

Grass

Texture Layer	Tiling scale	Notes
default	4 – 7	the default 3-D grass models
grass_out	4 – 7	clumpier 3-D grass models
grass_test	4 – 7	thick, tall 3-D grass, and a few flowers and weeds
grass_spring_01	4 – 7	brown, dry-looking 3-D grass and a few flowers
grass_rus_01	5	lush green 3-D grass and tall flowers and weeds
grass_rus_02	5	the same, but without flowers

Dirt

Texture Layer	Tiling scale	Notes
ground_usa_01	3 – 5	with 3-D dirt clumps
ground_rus_01	4 – 5	the same 3-D dirt clumps as ground_usa_01
ground_spring_01	5	with different 3-D dirt clumps; allows much more hidden mud
ground_test	5	with 3-D stones
ground_snowy	5	with 3-D dirt clumps; wetness is ice; supports snow?

Sand

Texture Layer	Tiling scale	Notes
sand_pbr_02	2.5 – 5	with 3-D dirt clumps
sand_pbr_03	2.5 – 5	only the ground texture differs
sand_test	2.5 – 5	only the ground texture differs; looks more like dunes

Mud

mud	1.5	no 3-D objects
mud_flat	1.5	only the ground texture differs; is a bit less wet looking

Gravel

Texture Layer	Tiling scale	Notes
gravel_road	3 – 5	with 3-D stones
gravel_road_rus	3 – 5	no 3-D objects
gravel_rus_01	2.5 – 20	no 3-D objects
gravel_pbr_test	3 – 5	no 3-D objects

Rocks

Texture Layer	Tiling scale	Notes
rocks_rus_01	1 – 3	no 3-D stones
rocks_rus_01_snowy	1 – 3	no 3-D objects; wetness is ice; supports snow?

Stone

Texture Layer	Tiling scale	Notes
stone_spring	1	no 3-D objects; wetness is ice
stone_snowy	1	no 3-D objects; supports snow?
stone_test	1	no 3-D objects

Concrete

Texture Layer	Tiling scale	Notes
us_concrete	3 – 5	no 3-D objects
us_concrete_old	3 – 5	no 3-D objects
us_concrete_tile	3 – 5	no 3-D objects

Asphalt

Texture Layer	Tiling scale	Notes
asphalt_rus_01	3 – 5	no 3-D objects
asphalt_rus_01_old	1 – 5	no 3-D objects
asphalt_rus_01_snow	1 – 5	no 3-D objects; wetness is ice
us_snow_asphalt	3 – 4	no 3-D objects; wetness is ice

Ice

Texture Layer	Tiling scale	Notes
ice_01	3	no 3-D objects; wetness is ice; can see through to a second layer underneath
ice_02	3 – 4	wetness is ice; is breakable; see TBD

Snow

Texture Layer	Tiling scale	Notes
snow_layer	2.2	supports soft_snow and crust_snow; supports snow depth
snow_spring	?	looks like snow_layer, but behaves like a normal terrain layer

Zone Icons

The following icons can be used to mark zones in the game. In the table below, only the 40×40 icon image is shown. The 30×30 icon is similar.

Use the 40×40 icon name in the zone's `Icon 40x40` property value, and use the 30×30 icon name in the zone's `Icon 30x30` property value.

Travel Services

Icon Image	Purpose	30×30 Icon Name	40×40 Icon Name
	fuel station	fuelStationImg30	fuelStationImg
	garage	garageImg30	garageImg
	gateway (tunnel)	gatewayImg30	gatewayImg
	locked gateway	gatewayLockedImg30	gatewayLockedImg
	service hub	serviceHubImg30	serviceHubImg
	trailer shop	trailerShopImg30	trailerShopImg40
	upgrade station	upgradeCreateImg30	upgradeCreateImg40
	watchtower	watchTowerImg30	watchTowerImg

Objectives

Icon Image	Purpose	30×30 Icon Name	40×40 Icon Name
	contest	contestImg30	contestImg
	new task	taskNewImg30	taskNewImg40
	task	taskImg30	taskImg40
	subtask	taskSubImg30	taskSubImg40
	crafting area	craftCrafting30	craftCrafting40
	lost cargo	constructionCargoImg30	constructionCargoImg
	lost truck	troubleAltVehicleImg30	troubleAltVehicleImg

Cargo Areas

Icon Image	Purpose	30×30 Icon Name	40×40 Icon Name
	airport	airportImg30	airportImg
	bunker	bunkerImg30	bunkerImg
	drilling site	drillingSiteImg30	drillingSiteImg
	factory	factoryImg30	factoryImg
	farm	farmImg30	farmImg
	lumber station	sawmillImg30	sawmillImg
	storage	townStorage30	townStorage
	timber station	lumberjackImg30	lumberjackImg
	warehouse	constructionWarehouseImg30	constructionWarehouseImg
	manual loading	manualLoading30	manualLoadingMarker40
	manual unloading	manualUploading30	manualUploadingMarker40

Vehicles

Icon Image	Purpose	30×30 Icon Name	40×40 Icon Name
	trucks	allVehicleImg30	allVehicleImg
	scout truck	scoutVehicleImg30	scoutVehicleImg
	highway truck	highwayVehicleImg30	highwayVehicleImg
	heavy-duty truck	heavyDutyVehicleImg30	heavyDutyVehicleImg
	offroad truck	offroadVehicleImg30	offroadVehicleImg
	heavy truck	heavyVehicleImg30	heavyVehicleImg

Ruins

Icon Image	Purpose	30×30 Icon Name	40×40 Icon Name
	brick ruins	ruins30Bricks1	ruins40Bricks1
	brick ruins	ruins30Bricks2	ruins40Bricks2
	brick ruins	ruins30Bricks3	ruins40Bricks3
	metal ruins	ruins30Metals1	ruins40Metals1
	metal ruins	ruins30Metals2	ruins40Metals2
	metal ruins	ruins30Metals3	ruins40Metals3
	wood ruins	ruins30Woods1	ruins40Woods1
	wood ruins	ruins30Woods2	ruins40Woods2
	wood ruins	ruins30Woods3	ruins40Woods3

Miscellaneous

Icon Image	Purpose	30×30 Icon Name	40×40 Icon Name
	GPS marker	markerImg30	markerGpsImg
	living area	craftLivingArea30	craftLivingAreaMarker40
	construction blockage	constructionBlockageImg30	constructionBlockageImg
	border booth	constructionBorderBoothImg30	constructionBorderBoothImg
	hard hat	constructionHardHatImg30	constructionHardHatImg
	offroad 1	offroadAltImg30	offroadAltImg
	offroad 2	offroadAltAltImg30	offroadAltAltImg

Additional Icons

The Saber guide includes a few more icons for which only a 30×30 or 40×40 icon are available. Since none of these looks particularly useful, I didn't bother to document them.

Saber also documents a few alternative 40×40 icon names for which the icon is in an unoutlined style. These look worse than the other icons, so I see no reason to use them.

There are more icons available in the game, but without documentation from Saber, there's no way to discover their icon names.

There is no known way to create new icons and integrate them into the game.

Cargo Types

The following tables give the internal name used by the Zone Settings Editor for each of the cargo types. The number of cargo slots used by each is also listed.

Pallets

Cargo Type	Editor Name	Cargo Slots
cement	CargoBags	1
packaged sand	CargoBags2	1
fuel	CargoBarrels	1
oil barrels	CargoBarrelsOil	1
bricks	CargoBricks	1
cellulose bricks	CargoCellulose	1

Crates and Containers

Cargo Type	Editor Name	Cargo Slots
consumables	CargoCrateLarge	1
vehicle spare parts	CargoVehiclesSpareParts	1
service spare parts	CargoServiceSpareParts	1
drilling spare parts	CargoServiceSparePartsSpecial	1
secure radioactive container	CargoRadioactive	1
cargo container	CargoContainerSmall	2
special cargo	CargoContainerSmallSpecial	2
oversized cargo container	CargoContainerLarge	4
drilling equipment	CargoContainerLargeDrilling	4

Frames and Straps

Cargo Type	Editor Name	Cargo Slots
wooden planks	CargoWoodenPlanks	1
concrete blocks	CargoConcreteBlocks	1
concrete slabs	CargoConcreteSlab	2
metal beams	CargoMetalPlanks	2
metal rolls	CargoMetalRoll	1
small pipes	CargoPipesSmall	2
medium pipes	CargoPipesMedium	2
large pipe	CargoPipeLarge	4
small cabin	CargoForkliftCaravanContainer2	2
large cabin	CargoForklift	4 (requires flat bed)
sequoia trunk	CargoSequoia	5

Vehicle Sections

Cargo Type	Editor Name	Cargo Slots
BA-20 armored car (rusted)	CargoBA20	2
BA-20 armored car (wrecked)	CargoBA20Add	2
rail section	CargoRailway	5
rocket engine assembly	CargoRocketEngine	1
stage 2 rocket fuel tank	CargoRocketPart2	3
stage 3 rocket fuel tank	CargoRocketPart1	3
airplane fuselage part	CargoPlane	5 (requires wide bed)
wing wreckage with engine	CargoWing1	5 (requires wide bed)
wing wreckage	CargoWing2	4 (requires flat bed)

Logs

Logs require a special logs carrier of the appropriate size, and each load fills the carrier.

Cargo Type	Editor Name	Cargo Slots
short logs	CargoLogsShort	*
medium logs	CargoLogsMedium	*
long logs	CargoLogsLong	*

Appendix B: Hand Edit the Map Files

The SnowRunner Editor makes it easy to edit your map and immediately see your changes. However, you sometimes need to do something that requires more power than what the editor offers or that is simply easier done outside the editor. The following sections describe what you need to know in order to edit files outside the SnowRunner editor, using either a text editor or an image editor.

Material Layout Bitmap

The material layout bitmap is `named mtrl_layout.tga`. It has one pixel for each terrain block, where the pixel value maps to terrain features as follows:

- a non-zero value in the red channel maps to the first material.
- a non-zero value in the green channel maps to the second material.
- a non-zero value in the blue channel maps to the third material.
- a non-zero value in the alpha channel maps to the fourth material.

If multiple channels are non-zero, an earlier material has priority over a later material. If all channels are zero, the Editor defaults to the first material.

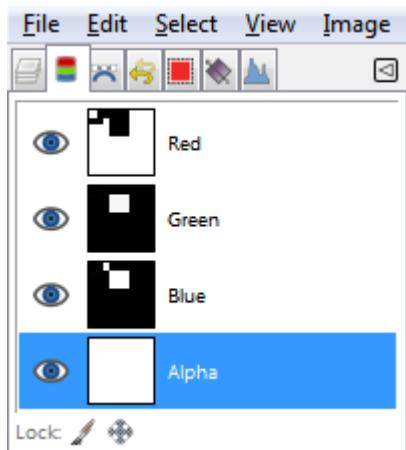
When painting material opacity in the Editor, the tint that the Editor applies to the terrain is taken directly from the R/G/B channels, so low values and/or conflicting values could be deceptive. The Editor ignores the alpha channel when tinting.

The editor has a bug in that it fails to put a value in the alpha channel when you paint opacity for the fourth material. Instead, it sets all channels to zero. This can be fixed in an image editor by assigning a non-zero channel to the alpha channel for all pixels. Since the fourth material has lowest priority, the Editor will correctly use the fourth material only when no other material is used.

Fix the Material Layout Alpha Channel in GIMP

It can be difficult to figure out how to use an image editor to edit the alpha channel while leaving the other channels alone. Here is the procedure that I use in GIMP.

- Open `mtrl_layout.tga` in GIMP. Because the alpha channel is zero everywhere, the image is transparent.
- Click the tab above layers to switch to individual Channels.
- Click each of the R, G, and B channels so that they are disabled.



- Click the paint bucket to fill the alpha channel with white. The entire image is now opaque.
- Select File->Overwrite mtrl_layout.tga

In the SnowRunner Editor, rebuild the terrain. Since the alpha channel is now non-zero, the fourth material is now correctly used where the R, G, and B channels are all zero.