Guide contents

Choose the authoring route

Build and load the first GRF

Inspect and release

Sources / References

Documentation: In progress **Quick start: In progress**

Offline copy from ModDocs. Source links remain clickable.

OpenTTD modding with NewGRF

OpenTTD's **NewGRF** format can add or modify vehicles, buildings, cargos, industries, rail types, objects and graphics. It is usually selected before starting a game; changing a NewGRF partway through an existing save may have unintended results. The OpenTTD NewGRF overview and usage guide describe the format and loading behavior.

> **Quick start:** pick one small visual or data change, follow the NML starter tutorial to build a `.grf`, enable it for a **new** test game, and verify the changed item. Do not begin by altering a long-running save.

Choose the authoring route

The development-tools page recommends **NML** for new authors: it is a high-level text language compiled to a `.grf`. NFO is lower level; YAGL is another route to inspect and rebuild existing GRFs. The file the game loads is the compiled `.grf`, not the NML source itself. A sprite-based project also needs original image assets and correct offsets.

Goal	Route	First test
Change a vehicle or industry property	NML definitions	Feature appears in a new game

Goal	Route	First test
Add sprites or a reskin	NML plus image files	Correct sprite at every zoom/state
Study an existing GRF	Source project or a suitable inspection tool	Confirm rights and dependency assumptions
Diagnose in-game graphics	NewGRF developer tools	Inspect sprite alignment and properties

Do not copy someone else's `.grf` and assume it is modifiable or redistributable. Prefer open-source examples and keep source assets under version control. The NML documentation is linked from the OpenTTD development tools page; verify its examples against your installed OpenTTD and NML versions.

Build and load the first GRF

Install a compatible NML toolchain and start from its smallest documented example. Set a unique GRF identity and a clear name before sharing it so it cannot collide with another project. Compile the source into a `.grf` and resolve the **first** compiler error rather than editing generated output. If the project references PNG sprites, check file paths and image format before adjusting offsets.

Place the compiled `.grf` in OpenTTD's `newgrf` content directory or use the game's content flow, then select it in the NewGRF settings **before creating the test map**. The usage guide explains manual installation and activation. Generate a new small world and verify the exact property or sprite you changed. If the GRF appears in the list but nothing changes, check that the feature exists in the selected climate/era and that another NewGRF is not overriding it.

Inspect and release

The NewGRF debugging guide covers in-game developer tools for sprite alignment and variables. Use them to diagnose alignment; always apply the measured correction back in the NML source, because in-game adjustments are not your release artifact.

Check multiple zoom levels and company colours when relevant. A data-only change still needs a new-game test for each feature it changes.

Package the compiled `.grf` with version information and a link to source or license terms. Test it from a clean OpenTTD profile with only declared dependencies. If the project uses someone else's sprites, obtain permission or substitute your own before publication. State the OpenTTD and NML versions used, plus whether saves made without the GRF are supported.

This guide is **unfinished**: the workflow is source-grounded, but it does not yet include a complete, locally compiled NML example with original sprites or a full compatibility matrix.

Sources / References

External project and documentation links cited in this guide. Check the upstream release and version before following version-sensitive steps.

- OpenTTD NewGRF overview
- usage guide
- development-tools page
- NML documentation
- NewGRF debugging guide