

Drut: Linting and Formatting for Cube Voyager Scripts

A Rust-based toolchain and VS Code extension for travel-demand model control-statement files

Pukar Bhandari

pukar.bhandari@outlook.com

2026-08-20

Table of contents

1 What it does	1
2 Three ways in, one engine	2
3 Install	2
4 Documentation	2

Under active development

Drut is a work in progress. If you hit a bug, please open an issue.

Travel-demand models are programs. In much of the U.S., the language they are written in is Cube Voyager's control-statement script (`.s` and `.block` files), and the tooling for it is thin: usually no syntax highlighting, no error checking until the model actually runs, and no consistent formatting across a team. A missing `ENDIF` can cost you a full model run to find.

Drut is my attempt to give those scripts the tooling that every mainstream language takes for granted.

1 What it does

- **Syntax highlighting** from a static TextMate grammar, so it works even before the language server starts. Control words, function calls, `keyword=value` pairs, data references (`MI`, `MW`, `DBA`, `ZONES`) and user variables are each independently recolorable.
- **Structural diagnostics**: unmatched `IF/LOOP/RUN/PROCESS` blocks, unclosed comments and similar real mistakes, flagged as you type, before you run the model.
- **Formatting**: consistent indentation and, optionally, casing for control words, pair keywords, data references and function names, on save, on paste or from the CLI.

- **Hover, autocomplete and “did you mean” spell-check** for control words and keyword names, plus **code folding** for every block kind.
- **Shared project configuration** through a `drut.toml`, so a team does not have to agree on CLI flags or editor settings one by one.

A formatter that rewrites model code has to be easy to hold back, so a range can be excluded with markers. Everything between them is left exactly as written:

```
; FMT: OFF
... a hand-aligned section you want to keep as is ...
; FMT: ON
```

2 Three ways in, one engine

Surface	For
<code>drut</code> CLI	scripting and CI checks
VS Code / Open VSX extension	day-to-day editing (it is a Language Server Protocol server, so any LSP-capable editor can use it)
Model Context Protocol server	AI coding assistants

All three are built on one shared Rust engine, so the editor, the CI check and the assistant always agree on what is valid.

3 Install

The extension resolves a working `drut` language server automatically on first activation:

- **VS Code:** Drut for Cube Voyager on the Marketplace.
- **Cursor, VSCodium and others:** the same extension on Open VSX.
- **CLI only:** build from source with `cargo build --release -p drut-cli` and put the binary on your `PATH`.

4 Documentation

The full user guide covers install, getting started, the CLI, editor and MCP references, formatter behavior and a field-by-field `drut.toml` reference.

[Open docs GitHub](#)

Drut is dual-licensed under MIT or Apache 2.0, and the code is on GitHub.