Abstract - TinyHexa

Ziel und Ergebnis

TinyHexa ist ein rein in C entwickelter, tastaturgesteuerter Hex-Editor für das Terminal. Er lädt beliebige Dateien, zeigt sie als Tabelle mit Offset-Spalte an und erlaubt bytegenaues Editieren in drei Perspektiven (Hex, Binär, ASCII), inklusive Umschalten per Funktionstaste und Statusanzeigen in Top-/Bottom-Bar. Damit erfüllt das Projekt das gesteckte Ziel einer schlanken, robusten Byte-Inspektion und -Bearbeitung ohne GUI-Overhead unter Linux und Windows. Die Implementierung folgt einem modularen Entwurf mit klaren Verantwortlichkeiten.

Von der Planung zur Umsetzung

Ausgangspunkt war ein Plan mit drei Programmphasen (Startup–Active–End), einem MVC-Model Zuschnitt (Model: Datei-/Speicher, View: TUI-Komponenten, Controller: Eingabelogik) und PDCurses als Bibliothek. Während der Umsetzung stellte sich jedoch heraus, dass im MSYS2-/UCRT-Umfeld eine vollwertige ncurses-Variante verfügbar ist. um Build-Systeme und Verhalten auf beiden Plattformen zu vereinheitlichen, wurde daher bewusst auf ncurses umgestellt. Außerdem wurden zu ambitionierte Nebenfunktionen (z. B. Suche/Hilfe) vorerst gestrichen, um die Kernfunktionalität stabil zu liefern. Die drei Phasen blieben erhalten, wurden aber praktisch so konkretisiert, dass Startup und End in main.c stattfinden, während die eigentliche Steuerung im Main Controller gekapselt ist.

Schlüsselmomente der Entwicklung

Eingabe-/Anzeige-Strategie statt If-Ketten. Bei der Darstellung kam es zunächst zu einer „Explosion“ an Fallunterscheidungen. Durch Einführung des Strategy-Patterns (Display_Strategy mit HEX_STRATEGY/BIN_STRATEGY) wurden Ausgabe und Formatierung pro Modus als virtuelle Funktionen gekapselt. Umschalten bedeutet somit nur noch das Austauschen des Strategy-Pointers.

Object-Pattern in C. Module wie Main_Controller und Main_Window verwenden konsistente Init-/Deinit Konstruktoren/Destruktoren und kapseln ihren Zustand in einer eigenen Struktur. Funktionen erhalten ausschließlich die Daten, die sie benötigen, per expliziter Parameterübergabe (Zeiger auf die jeweilige struct), ohne globale Variablen.

Pragmatisches MVC. Das Pattern trennt Model, View und Controller. In klassischem C stieß eine vollständige Entkopplung von View und Controller jedoch an Grenzen.

Benutzerfluss und Abschlusslogik. Main kümmert sich um den Dateipfad, startet oder beended die TUI nicht selbst (das passiert über den Controller/View) und fragt nach Verlassen der TUI im „normalen“ Terminal sichtbar und robust, ob ungespeicherte Änderungen gespeichert werden sollen, inklusive finalem Status-Text.

Build & Portabilität. Reproduzierbare Builds via CMake+Ninja. Mit GCC, unter Windows mit MSYS2/UCRT. Die Wahl von ncurses statt PDCurses.

Kritische Reflexion & Ausblick

Für Version 2.0.0 werden drei Schwerpunkte verfolgt.

1. Die Architektur schärfen. Geplant ist ein hybrider Zuschnitt, bei dem View und Controller enger verzahnt sind und gemeinsam auf ein klar abgegrenztes Model zugreifen. Das Strategy-Pattern wird konsequent als Erweiterungssteckplatz genutzt, um Verzweigungslogik weiter zu reduzieren und Erweiterungen einfach zu halten.
2. Die Funktionalität erweitern. Priorität haben UTF-8-Support über ncursesw, Such-/Sprungfunktionen, eine konfigurierbare Bottom-Bar, robustes Resizing sowie ein unmittelbares Eingabefeedback.
3. Aus TinyHexa wird in Version 2.0.0 NeoHexa, ein moderner TUI-Hex-Editor in Neon Farben für die Linux-Community, bereitgestellt als AUR-Paket. Die Neo-Optik lässt sich mit ncurses über Farbattribute schnell umsetzen. So adressiert das Programm eine Community, die gern schlanke, von Privatentwickler:innen erstellte Tools ausprobiert. Terminal-Apps im Neo-Stil sind derzeit sehr gefragt, NeoHexa kann dort bewusst anknüpfen.

Insgesamt zeigt das Projekt: Ein terminalbasierter, modularer Editor in pure C ist gut realisierbar, entscheidend ist jedoch die architektonische Struktur. Strategien statt If-Kaskaden und pragmatische, auf C zugeschnittene Design Pattern, führen zu besserer Lesbarkeit, höherer Erweiterbarkeit und geringerem Wartungsaufwand.