SubKilo: A Function Library Based on Kilo

Eingaeph

November 5, 2018

1 Motivation

Many programs would benefit from having a built-in default configuration file which would be used if an external configuration file was not found.

Some additional benefit might accrue if, as a non mandatory option, during startup the default configuration file could be edited with a few keystrokes, without exiting the program. If nothing else this would lessen those problems associated with a missing configuration file.

2 Kilo Is a Sound Base For This Work

In this light we seek the functionality of a built in editor, but contained in a C language function library. Sadly, we must write our own library, which is no simple task. Our library contains some 3 dozen functions, and numerous assert statements to check that everything fits together.

The prototype editor kilo is a suitable base for this work. Kilo is written in C and has no dependencies other than standard gnu/linux/gcc libraries, and the common features of unix/linux standard input and output. As a consequence a function library based on kilo can be small but capable. Our library is compiled from some 1500 lines of (C) code, and contains some 3 dozen functions.

3 Rewriting Kilo Is Justified

Kilo was written rapidly, in 2 or 3 days according to the author's notes, as a demonstration not a finished product. Kilo is remarkably capable, remarkably usable and remarkably free from bugs, but kilo has that first draft feel of execution path logic mixed up with more substantive calculations. Mixing excution path logic with other calculations often violates the first rule of function writing "do one thing based on a small bit of input and return without causing side effects". So we rewrite a lot and use functions to hide a clutter of assert statements, which we love.

4 Rewriting Kilo Is Not Justified

The functionality being sought can be gotten by starting an unmodified editor as a child process, etc. etc. When we learn of someone using that in a short piece of code we will reconsider whether linking functions or piping data from a child process is preferable. Right now we are guessing, actually, about how to do this with proper attentio to economy.

5 Build Steps

Subkilo, a function library based on kilo, is written in a number of steps. Each step involves creating a library based on a small number of functions, together with a test program linking them together.