LedgerSMB Coding Standards

The LedgerSMB Development Team

August 30, 2006

Contents

1 Introduction

Consistant coding style contributes to readability of code. This contributes in turn to fewer bugs and easier debugging.

While consultants which implement custom solutions for their customers are not required to follow these coding guidelines, doing so will likely contribute to efficiency of the project and, if desired, the likelihood of having the changes accepted in the main source tree.

2 File Organization and Whitespace

Files should be organized according to three principles: atomicity, performance, and readibility. While there is no maximum file size, there is no reason to group large amounts of code together when only a small subset of that code will be executed on any given run of the program. Similarly, core API for a single data structure entity may be grouped together even if each run may touch only a small part of the code so that these functions can be maintained together in a logical way.

Nested blocks of code should be indented with a single tab. This way, programmers can adjust the tab width of their editors to their preferences.

Lines longer than 79 characters are not handled well by many terminals and should generally be avoided. Continued lines should be indented by one more tab than either the line above or below (i.e. if the line above is indented by two tabs and the line below by three, indent the continued segment by four).

Lines longer than 79 characters cause problems in some terminals and should be avoided.

3 Comments

In an ideal world, the code should be sufficiently readible to be entirely understandable without comments. Unfortunately such an ideal is rarely attained. Comments should be used to annotate code and add information that is not already a part of code or programming logic itself.

Comments may include arguments and return values of functions (for easy reference), a summary as to why a particular design choice was made, or a note to alert future programmers that a particular chunk of code deserves additional attention. Comments should not, however generaly tell what the program is doing or how it does that. If such comments are required, it is a good indication that a block of code requires rewriting.

Additionally it is a good idea to provide a brief description at the top of each file describing, in general terms, what its function is.

4 Function Organization

Functions should be atomic. While there is no maximum length to functions, long functions may be an indication that a function may be non-atomic.

In general, when more than one line of code is being copied and pasted, it should instead be moved into its own function where it can be called by all entry points.