Programmer's Style Guide

A guide to writing usable and maintainable programs

Edition 0.0.0, 5 October 2002

Reinhard Müller

Table of Contents

Introduction

People often tend to judge the quality of software only by the number of it's functions, the beauty of it's user interface or the ease of it's utilization. But this is not the whole truth. Good software is more than that:

Good software is stable.

Well, actually this should be obvious: good software should have few bugs. (Oddly enough, some big and well-known software companies seem to be quite successful while not following this rule.) Of course, it is not possible to write 100% bug-free programs, but following a good coding style can save you from quite a few headaches.

Good software is fault tolerant.

This is even a higher aim than stability: not only that a program should not have bugs itself, but it should also be as tolerant as possible of failures of linked systems. A library function should always behave as reasonable as possible even when it is passed invalid parameters, and programs should give correct and meaningful error messages when the underlying software or hardware does not act like it should. Achieving this is not *that* hard, when you keep a few basic rules in mind.

Good software is reusable.

Some people think that reinventing the wheel over and over again is a basic requirement for a working software industry. However, other people don't. They try to build their code in a way that parts of it can be reused by other projects. But there are a few things to consider when you want your code to be *really* reusable.

Good software is maintainable.

Maintainability is not a question of simplicity. Even complex and powerful programs can be wonderfully maintainable, if you only take the necessary measures right from the start.

Good software is understandable.

What? Not only the original author of a program should be able to understand it, but everybody else looking at the code, too? Hey, that can't be right! That would mean that others can learn from my code! That others could even change my code, improve it, fix bugs, take over maintenance ... but wait! Couldn't that be a good thing, too?

Of course, there are a lot of things to do when you want to write good software. One of these things is that you should follow some standards in your coding style. To propose such standards, is the purpose of this guide.

Chapter 1 [How to Implement], page 3 deals with the proper way of implementing functions, not regarding naming conventions, code formatting and the like. Following the rules from this chapter will change how your software behaves, and make it more stable and fault tolerant.

Chapter 2 [How to Code], page 4 contains recommendations on how to translate an algorithm into code of a computer language. Whether or not you follow these guidelines (which are about modularity, naming conventions and all that sort of stuff) will not necessarily have impact on the functionality of your software, but on the reusability and the maintainability.

Chapter 3 [How to Format], page 6 gives you hints on the formatting of the code. This section deals with all the parts of the sourcecode that the compiler actually ignores, but help to make your software maintainable and understandable, like comments, whitespaces and indenting.

This text mainly concentrates of writing software in the C programming language, but most of the principles explained here can easily be translated to other languages.

This is not a replacement for the GNU Coding Standards (which you can find at `http://www.gnu.org/prep/standards.html`). This document does not deal with the specific issues about writing free software, and the very specific issues about writing software that should be incorporated into the GNU system. However, the above arguments probably make clear that only free software can be *really* good software, and therefore it is recommended that you read the GNU Coding Standards, too. In any case this document tries to not contradict the GNU Coding Standards.

This document is the result of a free documentation project. You can improve it if you want. Please look at `http://www.freesoftware.fsf.org/style-guide` for more information about this project.

1 How to Implement

1.1 Defensive programming

While it is good to believe that there is good in everybody, it is better to not rely on that.

Virtually every piece of code depends on underlying systems, either soft- or hardware. Software can have bugs, hardware can fail. Even your own code can have bugs. You can't do anything against that. But you can reduce the impact bugs and hardware failures have on the system.

1.1.1 Check the return values of the functions you call

Many functions provide return values that tell about success or failure of the function. If a function fails, react reasonably. Ignoring return values from a function can be like Russian roulette.

1.1.2 Check the validity of parameters in your functions

Every function you write (especially the external ones) carry the risk of being called with invalid parameters. A good function always behaves correct, even when it is called incorrectly.

If you use the glib library, the macros `g_return_if_fail()` and `g_return_val_if_fail()` provide a good way of checking parameters.

But what is correct behaviour when parameters make no sense? The answer is clear:

1.1.3 Return information about failure from your functions

Provide a means for the caller of your function to perceive failure. Make it possible to distinguish between different reasons of failure, if it could make sense for the caller to react differently. If your function fails, do not (besides the error code) return something that could look like a reasonable result (for example, if your function should return a pointer to a newly-allocated dynamic variable, always return a NULL pointer if it fails. Never return a random pointer, or a pointer to a improperly initalized data structure).

1.1.4 Expect the impossible

In a `switch` statement, always use `default`. If only specific cases are valid, place an assertion after the `default`.

If you use the glib library, use `g_assert_not_reached()`.

2 How to Code

2.1 Modularity

2.1.1 Be aware of the two levels of modularity

A program consists of different libraries, and a library consists of different object files (built from different source files). The following is not only true for the relationship between libraries, but also for the relationship between the different sources of a library, as well as the different sources of a program.

2.1.2 Make independent modules

If every module can be understood without reading the source of the other modules, people will understand the whole program better. If every module can be tested without relying on other modules, your code can get more stable. And, if every module can be debugged without digging through all the other code, you will be able to fix bugs much faster than otherwise.

Avoid circular dependencies in your modules. Avoid dependencies on too many other modules in a single module. Avoid modules that are tied together too strong.

2.1.3 Be clean in the headers

For C, the header files are the faces of your modules. If you put something in a header, you have to expect that people rely on it. Don't put implementation specific stuff in a header. Don't define structures in your header - define the structure in the main source, and simply put a type definition in the header, to make your structures opaque.

As every source file is a 'low-level module', every source file must have it's own header if it exports symbols that other source files of the same 'high-level module' (library or application) uses. Don't write a 'big' header file in which you define all symbols that are shared among your library or application, as that would make the internal dependency structure of your modules very unclear.

And, of course, protect all headers against multiple including.

2.1.4 Be clean with what you include

The `#include` preprocessor directives are a way of documenting dependencies. Don't include what you don't need. Explicitly include every header you directly depend on, even if it's implicitly included in another header.

Be careful to use `#include<...>` for system headers and headers that are external to your project and `#include "..."` for your own headers.

In header files, only include other headers when the code in this header needs the other header. Foreign headers you include in the header of your library have to be present on every system where your library should be used.

In source files, include all headers the source depends on, even those already included in the source's own header. This documents clearly what your source depends on.

2.1.5 Use the right order for includes

If your project has a global configuration file (like autoconf's `config.h`), this must be included in every source file, and it must be included as the *very first* line of your code after comments, so that all other header files can react on the defines. `config.h` may only include `#define`'s. Nobody expects code in such a file. Don't include `config.h` in a header file, unless you want to force all projects that use your header to have a `config.h`, too.

Next should be your source's own header file (where this source exports its external symbols). By putting no other includes before this, you implicitly check whether your header file is self-contained, i.e. if it contains all `#include`'s it needs to compile.

Then, include all needed header files that are external to your project, with the most usual ones first. Last, include the needed header files of the other modules of your project.

Never put any code before the `#include`'s. Nobody searches them somewhere else as at the very top of your source or header file.

2.1.6 Don't put code in header files

Nobody expects *real* code in files that end in `.h`.

2.1.7 Provide test code

When you write a module, you will write code to test it. That code is a part of the module. Put it in it's own file, document it, and make it a program that others can use to test if the module behaves correctly on their system. GNU Automake provides a very good means for running automatic tests when a program is built with `make check`.

2.2 Symbol names

2.2.1 Define and use a module prefix

Choose a prefix for a module, and use that prefix for all symbols in that module. If the module prefix is '`foo`', then all public symbols of that module should start with '`foo_`', and all private symbols with '`_foo_`'.

2.2.2 Name functions after the structure they operate on

If the module foo defines a structure bar, name the structure tag '`_foo_bar`', define the structure tag in the main source, and define a type '`foo_bar`' in the header. Name all the functions operating on this structure beginning with '`foo_bar_`'. For example, name a function that frees the memory of the structure, '`foo_bar_free`'.

2.2.3 Make your private global symbols static

Well, that's the reason why we have static symbols anyway, isn't it?

3 How to Format

text goes here

Appendix A GNU Free Documentation License

Version 1.1, March 2000

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other written document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

   The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

   The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

   A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgments" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgments and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover

Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgments", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

A.1 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.1
or any later version published by the Free Software Foundation;
with the Invariant Sections being list their titles, with the
Front-Cover Texts being list, and with the Back-Cover Texts being list.
A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being *list*"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.