Project Coding Style Research

Fletcher Gornick

September 27, 2021

Before giving my novice opinion on the Google C++ Style Guide, I thought I'd take a look at the meta surrounding it. From what I've found, there are very mixed thoughts on Google's approach to how C++ code should be written. It seems as though the public consensus is that this style guide is focused most on making code more compatible with Google's existing code base. That matter aside, I do believe that the guide is actually very insightful with only a few minor exceptions. My short answer as to if I'll use Google's C++ style directly, is no, I'll definitely use the guide as a sort of template to guide my process, but different projects should be approached in different ways. Although following one format for projects can lead to a very flexible and extensible program, it's much better to approach a problem your own way, this is what ultimately leads to strides in the development of new technology. But for the most part, I'll probably loosely follow the guide supplied by Google. With that in mind, I'd like to mention a couple topics that I, along with many others, don't necessarily agree with.

The first topic that I think should be mentioned is Google's take on exceptions. I'm sure most other kids on this assignment are bringing up the same point, but it's definitely something that can't be ignored. It was on a Quora post by Brian Bi where I found information about Google's take on exceptions [Bi, 2017]. Essentially, their take is that you shouldn't use them because of their own code base, they say it would be a hassle to introduce exceptions to their existing exception-free code. Now while they may be right that introducing exceptions now on their humungous project would be quite difficult, for anyone starting their own project, exceptions are a great way to debug errors in your program, and should definitely be adopted.

There are a couple other points the style guide goes over that I most likely won't use myself. I'm not all that knowledgeable on these topics though, so this will be more on a matter of personal preference. The style guide discourages multiple inheritence because of the possibility of complicated coad and overhead. They may be right, I'm just a silly little Junior after all, but I know that the most basic C++ library (iostream) uses multiple implementation inheritence. So while

I'm not 100% sure I'll even reach a point where I might consider multiple inheritence, I definitely won't let the style guide influence my decision on this point. The final point I'll be going over is the standard header guard. The question on why we use the standard header guard as opposed to the pragma was brought to my attention by a piazza post [Reinitz, 2021]. I think it's an interesting choice to use the `#pragma` method, and while I kinda doubt I'll be using it on my project, I do think it's a point that should have maybe been brought up in the style guide. But with all this in mind, I think the style guide is actually very helpful and definitely gets a bad rap for little reason, but I'm definitely not treating it like some sort of holy text. Stack Overflow is what I always refer to for good code practice, and I'll probably just stick to that, because it has many people to keep eachother in check. In conclusion, I'll just do shit my own way and I apologize for writing too much again (and for swearing).

References

[Bi, 2017] Bi, B. (2017). Why isn't it good to follow google's c++ style guide? https://www.quora.com/Why-isn%E2%80%99t-it-good-to-follow-Googles-C++-style-guide. Accessed: 2021-09-27.

[Reinitz, 2021] Reinitz, T. (2021). Header guard type question. https://piazza.com/class/ktacb43ye2wgh?cid=36. Accessed: 2021-09-26.