Knowledge article

General naming convention & few coding style guidelines

Created 1 year, 3 months ago Active 7 months ago Viewed 73 times 2 min read

c c++ code-quality

Share article ••• Edit article

General naming convention & few coding style guidelines

6

I have been doing day-to-day code reviews for more than a decade now. The benefits of code reviews are plenty, someone spot checks your work for errors, they get to learn from your solution, and the collaboration helps to improve the coding standards across product/project. For some of the code reviews where the changes/implementations are just for proof of concept kind of work and which often leverage the open source stack/code/sample codes, I often see the changes that are poorly written/commented and more often than not I get to hear that, this is "just for POC". Even so, I always feel there is something bare minimum that could be done to make look your changes look in right order and style, the least! Here are few of the thing I emphasize upon and remember these are not elaborate but minimalistic ones which could be followed.

1. Use names that describe the ***purpose or intent***
2. Be persistent with naming, if you use first letter as capital and subsequent letters as small, maintain that throughout
3. Variables naming, below are few options, idea is to be consistent across your project space a. Follow ***Camel*** casing i. For example : unsigned int Delay_count = 0; should be written as unsigned int DelayCount = 0; OR b. Another option is also that the names of variables (including function parameters) and data members are all ***lowercase***, with underscores between words. i. For example : unsigned int table_name = 0; OR c. Another option is also that the names of variables starting ***lower case*** and the ***first change*** in word can start with ***capital*** i. For example : unsigned int tableName = 0;
4. Its recommended to have 'p' or 'gp' suffix for the pointers defined, this does help in long run!! a. Local pointers to have p and global pointers to have gp, for example i. Local pointer "char *pData = NULL"; ii. Global pointer "char *gpData = NULL";
5. When declaring pointer types, the asterisk *() should be placed next to the variable name, not the type.This is emphasized from code readability per se than anything else a. unsigned*

int pData or unsigned int* gpData; --> This is not recommended way, do avoid b. unsigned int *pData or unsigned int *gpData; --> This is recommended way If you cant follow this, one can atleast try being consistent across code lines and files.

6. Function Names a. Functions to have ***mixed case*** and naming convention similar to variables could be followed i. Preferably functions should start with a capital letter and have a capital letter for each new word. 1) AddTable() 2) DeleteTable() 3) GetTablePointer()
7. Filenames *.c or *.h should be all ***lowercase*** and can include underscores '_'. Follow the convention that your project uses, there is a need to be consistent approach a. #include "Data.h" --> This should be avoided b. #include "data.h" --> This is preferred
8. In general macros should not be used.(There are strong arguments against use of same). However, if they are absolutely needed, then they should be named with all CAPITALS and underscores. a. For example : #define ROUND(x) …….. #define PI_ROUNDED 3.0
9. Comments a. Comments are absolutely vital to keeping our code readable. While comments are very important, the best code is self-documenting. Giving sensible names to types and variables is much better than using obscure names that you must then explain through comments. b. When writing your comments, write for your audience: the next contributor who will need to understand your code. Be generous — the next one may be you! c. Comment Style i. Use either the // or /* */ syntax, as long as you are consistent. ii. Preferred commenting is "//" since block comments could sometime erroneously comments required lines of code, this could be bit of time consuming but in long run could turn helpful

Reference

Coding style as such is a difficult topic to enforce but in general if we all can follow one style across same project, it would vastly help to maintain code, make it readable and much more!! Almost always, I refer this for any doubt I get while code implementation or code review [naming-style-guild-from-google]

created Nov 5, 2021 at 5:01

mahesh-gaikwad
753 1 11