

Whitespace \leq Comments \ll Code

WALTER E. BROWN, PH.D.

< webrown.cpp @ gmail.com >



Edition: 2017-09-28. Copyright © 2017 by Walter E. Brown. All rights reserved.

A little about me


- B.A. (math's); M.S., Ph.D. (computer science).
- Professional programmer for over 50 years, programming in C++ since 1982.
- Experienced in industry, academia, consulting, and research:
 - Founded a Computer Science Dept.; served as Professor and Dept. Head; taught and mentored at all levels.
 - Managed and mentored the programming staff for a reseller.
 - Lectured internationally as a software consultant and commercial trainer.
 - Retired from the Scientific Computing Division at Fermilab, specializing in C++ programming and in-house consulting.
- **Not dead — still doing training & consulting. (Email me!)**



Copyright © 2017 by Walter E. Brown. All rights reserved.

2

Emeritus participant in C++ standardization

- Written 125+ papers for WG21, proposing such now-standard C++ library features as `gcd/lcm`, `cbegin/cend`, and `common_type`, as well as the entirety of headers `<random>` and `<ratio>`. 
- Influenced such core language features as *alias templates*, *contextual conversions*, and *variable templates*; working on *requires-expressions*, comparison operators, and more!
- Conceived and served as Project Editor for *Int'l Standard on Mathematical Special Functions in C++* (ISO/IEC 29124), now incorporated into C++17's `<cmath>`.
- Be forewarned: Based on my training and experience, I hold some rather strong opinions about computer software and programming methodology — these opinions are not shared by all programmers, but they should be! 😊

Copyright © 2017 by Walter E. Brown. All rights reserved.

3

An opinion

[Commenting] is one of those topics that's briefly mentioned in an introductory programming class, and rarely or never again discussed [in a later class].

Someone should really bring it up with more experienced programmers.

— NAME WITHHELD BY REQUEST

Seems worth at least a lightning talk somewhere.

— YOURS TRULY

Copyright © 2017 by Walter E. Brown. All rights reserved.

4

A commenting classic

- “A famously bad comment style:

```
i = i + 1;          /* Add one to i */
```

- “There are worse ways to do it:

```
/******  
 *  
 *          Add one to i  
 *  
 *****/  
i = i + 1;
```

- “Don't laugh now, wait until you see it in real life.”

— ROB PIKE

Copyright © 2017 by Walter E. Brown. All rights reserved.

5

Excuses

- “I don't have time now to think about the comments.”
- “I'll put in the comments later.”
- “We don't have time for code reviews here.”
- Programmers in a hurry:
 - No time for planning. Or for thinking.
 - Let's start coding NOW!
- But imagine a builder saying, “I have no time for blueprints — we're just going to start nailing lumber for your house.”
- So, how do we make things better?

Copyright © 2017 by Walter E. Brown. All rights reserved.

6

Primum non nocere (First, do no harm)

- Much of the skill in writing good comments is in knowing when not to write them.
- A comment has zero (or negative!) value if it's **wrong**:
 - It provides misinformation and distracts us, over and over.
 - It's a subtle, constant drag on our thinking.
 - Yet it often survives in a way that wrong code couldn't.
- Even a correct comment may be useless or distracting:
 - When it parrots what the code already says.
 - When it's quickly outdated/obsolete/stale.
 - When it's phrased impolitely or ungrammatically.
- A comment adding no value is waste: remove or rewrite!

Copyright © 2017 by Walter E. Brown. All rights reserved.

7

Does that really help?

- Yes, it really does help, much more than seems obvious.
- *Here, the code is clear enough that I don't need comments, but the comments are so unclear I'm glad the code is there to explain them.*
— REMY PORTER
- *Don't be afraid to discard work you know isn't up to standard. Don't save junk, just because it took you a long time to write it.*
— DAVID EDDINGS
- Do the math: once you excise the "junk," the average quality improves instantly!
- Then what?

Copyright © 2017 by Walter E. Brown. All rights reserved.

8

Each comment should add value

- Express as much as you can via code:
 - After that, anything you still want to express ...
 - May be a plausible candidate for a useful comment.
- “Name it, don’t explain it”:
 - Rename rather than explain a poor name.
 - Introduce assertions instead of commenting assumptions.
 - Don’t comment sections of long functions; do extract smaller functions whose names capture the former sections’ intent.
 - Rewrite bad code; don’t comment it! *// This is bad, I know -,-*
- *[A] common fallacy is to assume [that] authors of incomprehensible code will somehow be able to express themselves lucidly and clearly in comments.*

— KEVLIN HENNEY

Copyright © 2017 by Walter E. Brown. All rights reserved.

9

When does a comment add value?

- When it says something that code can’t convey.
- But let’s please avoid prayers in our code:

```
auto leapyears = new Array {  
    1900, 1904, 1908, 1912, 1916, 1920, 1924, 1928, 1932,  
    1936, 1940, 1944, 1948, 1952, 1956, 1960, 1964, 1968,  
    : : :  
    2044, 2048, 2052, 2056, 2060  
}; // please god let this be far enough??
```

Copyright © 2017 by Walter E. Brown. All rights reserved.

10

Let's also avoid comments that are ...

- ... apologies: *// Dear future me. Please forgive me.*
- ... laments: *// I have to find a better job*
- ... artworks: *// (͡° ͜ʖ ͡°)*
- ... Suess-like poetry:
 - // Replaces with spaces the braces in cases*
 - // where braces in places cause stasis*
- ... useful only every 75 years: *// Halley's comment*
- ... pop-culture references:
 - long long ago; // in a galaxy far far away*
- But then there's this, from the era of punched cards:
 - debug = true // eslaf = gubed*

Copyright © 2017 by Walter E. Brown. All rights reserved.

11

In brief

- We ourselves are always the first:
 - We are the first beneficiaries of our good comments.
 - We are the first victims of our bad comments.
- *If the code and the comments disagree, then both are probably wrong.* — NORM SCHRIVER
- The most useful comment is the comment that does not have to be written!
 - *E.g., your name and today's date — this isn't school; code repositories record such information automatically.*
- *You should only need comments when ... you need to warn readers about [some kludge], just as on a road there are only arrows on parts with unexpectedly sharp curves.*

— PAUL GRAHAM

Copyright © 2017 by Walter E. Brown. All rights reserved.

12

Recommended reading

- Eric Steven Raymond:
The Cathedral and the Bazaar (2000).
- Jeff Atwood: *Coding without Comments* (2008).
- José M. Aguilar:
13 Consejos para comentar tu código (2007).
- Kevlin Henney:
Comment Only What the Code Cannot Say (2009).
- Rob Pike: *Notes on Programming in C* (1989).
- Robert C. Martin: *Clean Code* (2008).

Copyright © 2017 by Walter E. Brown. All rights reserved.

13

And finally

```
//  
// Dear maintainer:  
//  
// Once you are done trying to 'optimize' this routine,  
// and have realized what a terrible mistake that was,  
// please increment the following counter as a warning  
// to the next guy:  
//  
// total_hours_wasted_here = 42  
//
```

Copyright © 2017 by Walter E. Brown. All rights reserved.

14

Whitespace \leq Comments \ll Code

FIN

WALTER E. BROWN, PH.D.

< webrown.cpp @ gmail.com >



Copyright © 2017 by Walter E. Brown. All rights reserved.