EMBRACING DOCUMENTATION: GOOD CODE IS ONLY HALF THE JOB

WEDNESDAY 02 OCT 2024

Today I'll be deviating from the usual technical deep dives. Let's talk about something less glamorous but perhaps the most important part of software development: **documentation**.

If you just groaned a little, you're not alone. As an undergrad, I had the same reaction. Writing code is fun (at least I hope you think so), but explaining it? Not so much. Yet, good documentation is what turns your code from "*my brilliant masterpiece*" to a useful tool for **others**. And trust me, you'd thank yourself later. Not soon, but probably six months later!

WHY BOTHER?

Communication is **everywhere**, whether you're a student working on a group project or a full-time software developer. With the rise of remote work and agile development, you may think this means less communication, but you'd be surprised.

You have to communicate your thought process to other developers, and the product you built still "communicates" with its users. That's where documentation comes in. It's like writing a letter to your future self — or the next poor soul who has to understand your code or product. Good documentation is clarity on a page: it's how you ensure your hard work doesn't go to waste because someone (*probably you*) couldn't figure out what your clever function was actually doing.

MY JOURNEY WITH DOCUMENTATION

Over the years, I've learned that great documentation isn't just a reference. It's a guide, a bridge, and most importantly, a lifesaver. The challenge? Balancing brevity with thoroughness. You should write as if you're teaching, not just telling. Start simple, then dive deeper as needed. And always get feedback from a fresh pair of eyes – a fellow developer, a friend, or even a user – they'll spot those "obvious" mistakes that aren't so obvious. If someone else understands what you're writing, end-users can probably understand your product, whether it's a user guide or developer guide.

Documentation isn't just an afterthought. It's something worth investing time into. A clear straightforward guide beats a fancy one. Great documentation doesn't just explain, it makes your product *accessible, understandable, and meaningful*.

KEYS TO EFFECTIVE DOCUMENTATION

Writing clear and effective documentation is a skill that benefits both technical and non-technical users. If you want to improve your documentation, focus on three key areas:

- **Know Your Audience:** Tailor the level of detail and language to suit either developers or end-users. Developers might need technical explanations and code samples, while end-users learn from simpler language and explanations.
- **Structure Your Content:** Use clear headings, navigation bars, and even a search function if you're creating a larger documentation site. This helps people find what they're looking for without getting lost.
- **Integrate Documentation into Your Process:** Documentation should be an ongoing practice. Regular incremental updates are less time-consuming than big overhauls and help keep information accurate.

Documentation is like unit testing—easy to think of as a chore, but ultimately worth it. A process for reviewing documentation regularly keeps it relevant, benefiting both developers and end-users in the long run. After all, what's the point of building a great product if people can't figure out how to use it?

THAT'S IT!

Hopefully, you're now inspired to write better documentation — maybe not ***inspired*** but at least do it well! You can take some inspiration from other companies' documentation by clicking the links below too!

Flutter GitHub aws stripe twilio slack

IMAGES

https://www.zegocloud.com/blog/programming-memes
https://medium.com/analysts-corner/the-art-of-writing-good-documentation-6e4ce4cd3126
https://programmerhumor.io/memes/documentation/
https://programmerhumor.io/programming-memes/read-the-documentation/
https://medium.com/@imranfosec/documentation-as-code-an-efficient-way-to-save-you-from-silly-questions-3de97299e5b8

BACK TO HOME