Clean Code: Be The Hero NDC Oslo 2023 Event Report

For my event of choice, I decided to watch Ben Dechrai's presentation "Clean Code: Be The Hero" at NDC Oslo 2023. The concept of clean code has interested me ever since I read "Clean Code: A Handbook of Agile Software Craftsmanship" last year. In his presentation, Dechrai makes a very compelling case for adopting clean code practices across the software development industry, to the benefit of developers and project managers. As someone who works with software both professionally and independently, much of what he discusses in this presentation resonates with my current work and future career goals. In this report, I will analyze the benefits, techniques, and challenges of implementing Clean Code guidelines in software development.

Dechrai begins by outlining four main benefits of clean code. He highlights improved productivity, reduced costs, enhanced software quality, and better testability as the main advantages. For improved productivity, he compares writing code to writing a book, if the "story" of the code follows a linear pattern, it becomes significantly easier to understand and expand upon. One section that particularly stuck with me was when Dechrai noted that while software was once written exclusively for computers, modern advancements in storage technology and computing now allows us to write code for human readers. In earlier days of software development, short variable names and abbreviated file names were necessary due to limited memory space. But today, with the wide adoption of cloud computing and storage, descriptive names, comments, and file structures are not only easily affordable but also standard when writing modern software. This shift means less time spent deciphering past work and more time focused on meeting deadlines and driving projects forward.

When it comes to reducing costs, Dechrai explains that readable code minimizes the time and resources required for debugging and maintenance, which is something that project managers are likely to value. However, I found myself not necessarily agreeing with his point; while clean code can streamline future expansions and assist debugging, the initial effort of making the software work can slow down the development process when there is too much focus on meeting guidelines. I wish Dechrai had spent more

time addressing this counterargument, as it seems to be a common concern among those who oppose strict clean code guidelines. On the other hand, his points on improved software quality and enhanced testability were especially interesting to me. Code that is easy to read is not only simpler to debug but also more straightforward to modify. Dechrai also observed that clear code often reveals errors through just reading alone, even before testing begins.

Moving on to the techniques of writing clean code, the presentation breaks these down into naming conventions, code formatting, code organization, and documentation. Naming conventions, which have become essential in modern programming, involve choosing clear, descriptive names for functions, variables, and classes. In the presentation, an example is shown of changing a function's return value name from "result" to "average", which might seem minor, but at scale, this clarity at scale can greatly improve the overall legibility of the codebase. Alongside good naming, proper code formatting also plays a very vital role. Dechrai demonstrated that breaking complex functions into smaller, more granular chunks not only helps with code comprehension but also can help reinforce the overall structure of the structure of the code. While it can be tempting to prioritize speed over structure, sacrificing these guidelines can undermine long-term efficiency. Additionally, the presentation warned against the overuse of comments. Although comments can help provide clarity to very complicated code, an excess may actually reduce legibility to perfectly legible code. My big takeaway from this section is that cleanly written code has no need for an abundance of comments, as the code itself should be written in a way that its intention is immediately obvious.

Despite the clear advantages of clean code, it does not come without its challenges. Dechrai identifies three main obstacles: tight deadlines, evolving requirements, and the increasing complexity of large codebases. The pressure of deadlines often forces developers to choose speed over cautious code writing, which can potentially sacrifice clarity for meeting deadlines. Additionally, as codebases grow and requirements change, updating a complex codebase to meet clean code practices can become a massive undertaking. This interaction with changing requirements and code complexity shows that the practical

challenges of maintaining a consistently clean code environment must be solved before actual development begins, as changing requirements can exponentially increase development time.

In conclusion, Ben Dechrai's presentation at NDC Oslo 2023 does a fantastic job of explaining the value of clean code in modern software development. While the benefits (increased productivity, reduced long-term costs, higher software quality, and improved testability) are significant, they come with challenges that cannot be ignored. The balance between writing legible code and meeting immediate deadlines, as well as managing ever-growing codebases, is tricky to do right. Nevertheless, the insights offered in this presentation have greatly deepened my understanding of clean code practices, and I will undoubtedly take much that I have learned from this presentation into my professional career going forward.

References

Dechrai, B. (2023, July 7). *Clean Code: Be The Hero - Ben Dechrai - NDC Oslo 2023*. YouTube. https://www.youtube.com/watch?v=Yu4kIgAH_NM

Martin, R. C., & O'Brien, T. (2021). *Clean code: A Handbook of Agile Software Craftsmanship*. Upfront Books.