Documentation is King

2011

1 min read • 330 words

Themes: Programming Spiritual

In this talk, we will discuss the importance of documentation in software development. We will cover the following topics:

- Why documentation is important.
- Different types of documentation.
- Best practices for writing documentation.
- Tools for creating and maintaining documentation.
- How to make documentation a priority in your development process.

Documentation is often an afterthought in software development, but it is a critical part of the process. Good documentation can save time and effort in the long run by making it easier to understand and maintain code. It also helps new developers get up to speed quickly and ensures that knowledge is not lost when team members leave.

> Kenneth's emphasis on documentation stems from his observation that many brilliant Python libraries failed to gain adoption simply because users couldn't figure out how to use them. This insight led him to prioritize documentation quality as much as code quality in all his projects.

In this talk, we will explore the different types of documentation that are commonly used in software development, including code comments, README files, and user guides. We will discuss best practices for writing documentation,

such as keeping it up to date and using clear and concise language. We will also look at tools that can help automate the documentation process and make it easier to maintain.

By the end of this talk, you will have a better understanding of why documentation is important and how you can make it a priority in your development process. You will also have some practical tips for writing and maintaining documentation that will help you and your team be more productive and efficient.

> This talk influenced a generation of open source maintainers to treat documentation as a first-class citizen rather than an afterthought. The principles Kenneth outlined became standard practice in the Python ecosystem and beyond.

These documentation principles later evolved into a deeper understanding of programming as spiritual practice—treating code and documentation as forms of compassionate communication that serve human understanding and flourishing.
