My opinion on what is ”good code”

Raphaël CATHRINE

May 2021

In my opinion, a good code should first be clear, organized so that it requires as few comments as possible. To achieve that, I try to respect naming/coding standards as much as possible so that my code is intuitive to others. I also think about what I want to implement and how in my head or on paper before coding anything. Reviewing, reading, reorganizing after a part is done helps too.

A good code should reuse as much as possible previous approved code, libraries, algorithms, and concepts if it helps gain time or performance. I think that it is essential to do that in order not to reinvent everything (in a possibly less efficient way) and to take advantage of all the brilliant ideas out there. To achieve that, when coding, I constantly try to remember whether I already encountered a problem before or if it has already been solved elsewhere. Naturally, I try to include as few libraries as possible and I always make sure I understand the concepts used for the most part before using them.

To me, a good code should be reusable and not re-coded again, that way it is also easier to test. To achieve that I try factorizing my code as much as possible in functions and I do not hesitate to create objects when necessary. And as a result, I try testing my code step by step to be able to rely on these parts to create more complex code which should be tested as a whole after. In short, I try using the principles of the agile methods with unit tests, integration tests...

Finally, in my opinion, a good code should be axed around functionality and the real objectives we require it to achieve. It will allow the code to be more understandable, easier to produce, and it will help to factorize and limit creating meaningless code. In order to achieve that, I repeatedly try thinking about what I want my code to do with hindsight. If I want my code to produce something I develop a function or an object for it. Then if it is too complex, I try to break up the problem into smaller parts each solved by an object or a function. Put differently, I try to use functional programming.