Slide 1

Slide 2

Most software systems are unnecessarily overloaded with almost unnecessary features, which impairs the usability of their use by end users and also complicates their support and development by developers. Following the KISS principle allows the development of solutions that are easy to use and maintain.

Slide 3

What is the KISS principle?

KISS is a design principle adopted by the U.S. Navy in the 1960s. The KISS principle states that most systems work best if they stay simple rather than get complicated. Therefore, in design, simplicity should be a key goal and unnecessary complexity should be avoided.

Slide 4.

There are many ways to decipher the acronym: 'keep it simple, stupid'. It is 'Keep it Simple, Silly', 'keep it short and simple', 'keep it simple and straightforward' and 'keep it small and simple'.

Slide 5

The term was reportedly coined by Clarence Johnson, lead military engineer for Lockheed Skunk Works (makers of the Lockheed U-2, SR-71 Blackbird and many other aircraft).

Slide 6

While it has been popular to use the decryption "Keep it simple, stupid. This principle is best illustrated by the story when Johnson handed a team of aeronautical engineers a set of tools, setting them a condition: a mid-level mechanic must be able to repair the jet they were designing in the field with just those tools. So "stupid" refers to the relationship between the fact that things break and the complexity of the repairs needed to make them.

Slide 7.

In programming, following the KISS principle can be described as follows:

don't use abstruse and non-obvious solutions, you must write code as simply as possible

You don't need to integrate a huge library if you only need a couple of functions from it;

there is no sense in increasing the level of abstraction to infinity, you must be able to stop in time;

You don't always need absolute mathematical precision or extreme detail - most systems are not built to launch space shuttles, data can and should be handled with the accuracy that is sufficient for a quality solution to the problem, and the detail should be provided as much as the user needs, not as much as possible.

Often the most correct solution is the simplest realization of a task which does not contain anything superfluous.

The simpler the code, the easier it is to understand, both for you and the others involved in maintaining it. By simple we mean that we don't use any tricks or complicate things unnecessarily.

Slide 8.

Examples:

Examples of violating this principle include writing a separate function just to perform an addition operation, or using a bitwise operator (right shift >> 1) to divide integers by 2.

(4 >> 1) === (4 / 2)

The latter may be more efficient for some software compilers than the usual division / 2, but it greatly reduces code comprehensibility.

Important note: For JS engines, the efficiency of the shift operation is negated by expensive integer conversions.

With this approach, you are doing clever coding and over-optimization. Both of these things will make your code less and less understandable in the long run, both to other developers and to you, because you may have to deal with it again in a month, two years.

Slide 9

KISS is a very general and abstract design principle that contains almost all the other design principles. Design principles describe how to write "good" code. However, what does good code mean? Some people think it's code that runs as fast as possible, some think it's code that involves as many design patterns as possible... But the right answer lies on the surface. Code is information in its purest form.

Slide 10

And the basic criteria for the value of information are 1) reliability 2)accessibility 3)understandability. It is clear why reliability and accessibility are important. The code is useless if it has bugs or if the server with the application is "dead". But why is the readability of the code important? It's easier to look for errors in the clear code and easier to change, modify and maintain it. So, readability is the main value the programmer must aspire to. But there is one pitfall here. The thing is that clarity is a purely subjective thing. There should be a more objective criterion of clarity. This criterion is simplicity. Indeed, a simple application is clearer than a complex one. But simplicity is difficult to achieve.

Slide 11

Here's what Peter Goodwin writes in his book The Craft of Programming:

If a project is simple, it's easy to understand... Developing a simple project is not so easy. It takes time. For any program, however complex, the final decision comes from analyzing a huge amount of information. When the code is well designed it seems as if it could not be otherwise, however it is possible that its simplicity is achieved as a result of much mental labor (and a lot of refactoring). Doing the simple thing is difficult. If the structure of the code seems obvious, don't think it was given without difficulty.

Slide 12

So, the KISS design principle proclaims that simplicity of code is paramount because simple code is the most understandable.

Almost all design principles are aimed at making code understandable. If you violate any design principle, you reduce the understandability of the code.

Slide 13.

Unintelligible code automatically makes a person feel that the code is complicated because it is difficult to understand and modify. If any of these principles are violated, the KISS principle is also violated. Therefore, we can say that KISS includes almost all of the other design principles.

Slide 14.

Design patterns describe the most successful, simple and straightforward solutions to certain problems. If you use a design pattern where there is no problem that the pattern solves, then you are violating KISS by making the code unnecessarily complicated. If you're NOT using a design pattern where there's a problem that matches the pattern, then again, you're violating KISS by making the code more complicated than it could be.

Slide 15.

Since different people have different ideas about such a thing as "simplicity", the following misconceptions about KISS have become widespread:

Misconception 1. If you think that simple code is the code that's easiest to write, then you could interpret that the KISS principle calls you to write the first thing you can think of without thinking about design at all.

Misconception 2: If we think that simple code is code that requires as little knowledge as possible to write, then we can interpret that the KISS principle calls for not using design patterns.

Slide 16.

Simplicity in this case should be understood as non-complex, devoid of artificiality, the most natural, uncomplicated, easily understandable.

Slide 17

Once again, back to history, we have already said that the principle of KISS originated in aircraft engineering and there are cases where too zealous workers nailed on the plane extra plates of armor to make the plane more survivable in battle, resulting in the weight of the aircraft more than calculated and the plane simply could not take off. In addition, the qualifications of many workers were low. Under such conditions, airplane designs that a drunken, unskilled worker could not assemble incorrectly, even if he wanted to, were of particular value. One echo of the design decisions of that time is the impossibility of mixing up and plugging the wrong plug into the socket inside the computer. However, if the result of an aircraft engineer's work is a drawing, according to which the product will be created, in the case of a programmer, the product is the drawing itself (figuratively speaking). In the case of the programmer he has to write the code so that a drunken unqualified programmer can make changes to it in accordance with the changed business requirements (that is to change the drawing, not to build an airplane). Because of the differences in the specifics of aircraft construction and programming, the "Keep it simple stupid" transcription, appropriate in aircraft construction, no longer reflects so well the essence of the principle for the programmer. Many lazy programmers decipher "keep it simple stupid" as "don't bother designing."

Slide 18/1.

Fortunately, KISS has some of the other transcripts I mentioned above, one of which, in my opinion, best captures the essence of KISS in programming - "keep it simple and straightforward". Straightforward means simple, honest, frank. "Keep it simple and straightforward," therefore, can be loosely translated as "Keep it simple and declarative," and to achieve declarativeness requires design.

Slide 18/2.

Use KISS in your work and we'll all be better off!