**Be Stupid and Lazy**

It may sound amazing, but you could be a better programmer if you were both lazier and more stupid.

First, you must be stupid, because if you are smart, or if you believe you are smart, you will stop doing two of the most important activities that make a programmer a good one: Learning and being critical of your own work. If you stop learning, it will make it hard to discover new techniques and algorithms that could allow you to work faster and more effectively. If you stop being critical, you will have a hard time debugging and refactoring your own work. In the endless battle between the programmer and the compiler, the best strategy for the programmer is to give up as soon as possible and admit that it is the programmer rather than the compiler who is most likely at fault. Even worse, not being critical will also cause you to stop learning from your own mistakes. Learning from your own mistakes is probably the best way to learn something and improve the quality of your work.

But there is a more important reason why a good programmer must be stupid. To find the best solutions to problems, you must always start from a clean slate, keeping an open mind and employing lateral thinking to explore all the available possibilities. The opposite approach does not work out so well: Believing you have the right solution may prevent you from discovering a better one. The less you know, the more radical and innovative your ideas will be. In the end, being stupid — or not believing yourself to be so intelligent — also helps you to remain humble and open to the advice and suggestions of your colleagues.

Second, a good programmer must also be lazy because only a lazy programmer would want to write the kinds of tools that might ultimately replace much of what the programmer does. The lazy programmer can avoid writing monotonous, repetitive code, which in turns allows them to avoid redundancy, the enemy of software maintenance and flexible refactoring. A byproduct of your laziness is a whole set of tools and processes that will speed up production. So, being a lazy programmer is all about avoiding dull and repetitive work, replacing it with work that's interesting and, in the process, eliminating future drudgery. If you ever have to do something more than once, consider automating it.

Of course, this is only half the story. A lazy programmer also has to give up to laziness when it comes to learning how to stay lazy — that is, which software tools make work easier, which approaches avoid redundancy, and ensuring that work can be maintained and refactored easily. Indeed, programmers who are good and lazy will look out for tools that help them to stay lazy instead of writing them from scratch, taking advantage of the efforts and experience of the open source community.

Perhaps paradoxically, the road toward effective stupidity and laziness can be difficult and laborious, but it deserves to be traveled in order to become a better programmer.

By [Mario Fusco](http://programmer.97things.oreilly.com/wiki/index.php/Mario_Fusco)