Opinionated

Adrian Kosmaczewski

2021-03-05

Programming is a very opinionated activity. Unfortunately, those opinions are seldom based on facts, and most of them are futile, and lead to stupid arguments on Reddit or Hacker News or the comments section of a blog.

Most of your conversations with fellow programmers will most probably revolve around one of the subjects enumerated below.

Classical example, the choice of the programming language. Most developers develop a religious attachment to their favourite programming language, and saldly are never able to go past this rather childish feeling. It is like refusing to leave your parents' home and move on with your life once you have the means to.

The truth is, programming languages are just a tool. JavaScript is great at some things, just like C++ is great at others. In some cases you cannot use JavaScript, and you have to use C++; sometimes it is the opposite situation. In some cases you can use several different programming languages to solve the same problem, in more or less the same amount of time, for example Ruby vs. Python. You might like or not the fact that Python uses indentation with specific semantics, but you cannot deny the power of its libraries. Similarly, Ruby is very approachable, but you might dislike that it does not feel fast or efficient.

The truth is that the choice of a particular language is an opinion; and as such, *it does not matter*. Seriously. Choose any language you want, as long as you and your team are comfortable with it, it has some decent prebuilt libraries ready to use, and it should work out just fine. Just do not enter futile arguments about how your choice is better than someone else's.

There is no "best", not even "better" programming language.

What is "better" is to go and learn a new language every year. I have been consciously been learning a new language every year for the past 30 years, and this has provided me with an invaluable tool: *context*. Being able to understand how languages are really different from each other, because you have actually tried them, is a gazillion times better than swallowing other people's misconceptions.

Open your mind; learn.

The same argument applies to IDEs, and more generally speaking, to text editors: Visual Studio, Xcode, AppCode, IntelliJ, Eclipse, Sublime Text, Vim, Emacs, Notepad++, EditPlus, Visual Studio Code, TextMate… they are all great and shitty at the same time. Each comes with a long list of relative advantages and an equally long list of terrible flaws that will be never fixed. You read right: never. They are just like that. Take it or leave it.

These tools will be your primary means to earn your bread during your whole career as a code monkey, so it is normal that you pay attention to them. You are going to be standing in front of one of those pieces of software for hours, days, weeks, years, and you are going to both love and hate them. All of them suck and shine in different ways; some are lean; some are heavyweight beasts. Some are fast. Some are slow. Some have awesome debugging capabilities but crash miserably. Some are great in embedded software development and others are better at web development.

And developers will argue, and even worse, they will openly dismiss, criticise, make fun and be arrogant in front of you, just because they have been never able to write anything useful or popular in any of those tools. There is one reason most programmers are so vocal about their preferred tool, and defend it to the end of times: because it is the only one they know how to use properly. As simple as that.

Take a Visual Studio user and put him in front of Vim. Take an Xcode user and try to make him use IntelliJ. Ask a Notepad++ user to switch to Atom. Watch their faces, and listen to the stream of incivilities.

Remember that they are just trying to hide their ignorance by criticising the tool they have in front of them. It is not that the tool is wrong in any way; it is that they just do not know how to use it.

In your case, remember to never, ever talk about something you do not know. Learn, use, try to make something with the tool; then make your own opinion.

The keyboard you choose also defines you somehow as well in the eyes of the "community", whatever that means. QWERTY vs Dvorak, blue versus cherry MX springs, ergonomic vs. compact, Bluetooth vs. USB, IBM Type M vs Apple Extended II.

The keyboard is the primary means of communication with the computer when you are a developer, and it is primarily used to communicate your ideas inside of the editor we discussed in the previous section. Nothing else. It does not define you in any way, it does not make you better, it does not make you more valuable. But of course a noisy keyboard in an open space office can make people hate you more; that, yes.

We programmers are human beings, and we too are drawn to simpler views of the world, just like anyone else; we are also caught in opinion storms, and we

also enter arguments that have absolutely no value.

You will probably (sadly) find those arguments interesting while you are young, but as soon as years start piling up on you, you will find them futile, and without substance. The important thing is to grow your own taste and knowledge; to try new things, to download new IDEs and languages, to play with them, and also to keep a track record of your discoveries.

Your taste will grow, and also your respect for the choices of others. It is only at that moment that you will be able to evaluate technology properly.