Programming technique

Marcin Pawlukiewicz & Sergio DeSousa-Rosa Jr.

One of the programming techniques that Sergio and I incorporated was frequent compiler checks in Rust. Rust places a strong emphasis on safety and correctness. Its compiler is very smart and provides valuable improvements. Not only does Rust catch errors, but we can also easily get rid of any type of warnings, as Rust tells us how to fix them. Therefore, we can create higher-quality code overall.

We believe that in every programming language, the compiler might be our friend. It is important to find a good one, and then it is essential to compile often. Once we have that, even if at first some of the errors are not clear, over time, as we gain practice, we will more easily know what we missed in our code and how to fix it efficiently. Even if some programming languages don't have as smart a compiler as Rust.

Two of the most helpful commands I found in Rust were:

- `cargo build` – it builds our projects, including all dependencies (external libraries). We will know if everything works properly once we run this command.
- `cargo check` – it is faster, a lightweight check of our code. This one doesn't generate the executable file. Other than that, it works similarly to `cargo build`.

We think, overall, it is not worth writing 200 lines of code without compiling, as the results might be dreadful. If the language's compiler is not intuitive, that can also lead to one error after another, taking forever to fix. Moreover, the errors might not be clear, as one error can produce a hundred new ones, and we will not be able to figure out that this one thing caused us such a headache. We will have ninety-nine out of a hundred errors that are incorrect, as they were created due to the original problem. We will end up scratching our heads for a couple of hours before we find the cause. The worst-case scenario is that you will start coding the same program from scratch because you got lost.

Compiling often has many benefits, such as early error detection, resulting in saving time and reducing the likelihood of introducing bugs. You will improve your code quality, as some compilers provide suggestions for writing cleaner and more idiomatic code, especially Rust. Lastly, this will turn into time efficiency, as debugging becomes more manageable.

In summary, frequent compiler checks are a great programming practice, and this is one of the techniques that every programmer should implement. By doing so, the coding journey becomes less painful, more efficient, and you will have extra time to go out. ☺