Why Prettier will make you a more productive programmer - Why I switched from ESLint to Prettier for formatting JS

Written by Seth Corker on Benevolent Bytes

Keeping code organized and looking prettier Photo by Sanwal Deen

Linting is a great productivity tool, tools such as ESLint can find potential errors and help foster a consistent style across teams. It is infinitely customisable although many developers, myself included, opt for some predefined rules. I choose the popular Airbnb ESLint rules to save time, it has a good style and a good reasoning behind why each rule is the way it is. Such as the reference rules, preferring `const` over `var`:

> Why? This ensures that you can't reassign your references, which can lead to bugs and difficult to comprehend code.

Taken from https://github.com/airbnb/javascript/blob/master/README.md

My problem with ESLint Once everyone in the team is onboard, you have a consistent style and less headaches. Or so you would think, the reality, however, is much more tedious and time-consuming depending on how you choose to enforce the rules.

One option of enforcing your ESLint rules, is with an iron fist. Everything is an error and the CI pipeline will fail if you don't conform. This approach will get the result you desire but might drive your team insane. You will have developers cursing whoever set up the rules and complaining, rightfully, why a missing dangling comma is a cause for a merge request sitting in limbo. So what's a fewer heavy-handed approach?

Another approach is to relax the rules and just encourage your team to follow the rules. This approach can work well depending on how much people care. In my experience, bug fixes near the end of the sprint mean the rules will go out the window. The end result is a mostly consistent code base with some occasional "refactor" commits when you discover some interesting deviations from the rules.

Both of these options are fine but there is an underlying problem here. Some rules are great, they encourage cleaner, more concise code. The formatting rules, however, don't change what is executed at runtime and more often than not, send developers into a rabbit hole of stylistic wrangling. Automated formatting tools work for some of the problem but they aren't very powerful and still require a helping hand most of the time. This wastes time and effort which could be better spent on something productive. This is where prettier comes in.

<AutoplayVideo src={[{ src: "s3-bucket://build/2018-04-28-why-prettier-will-make-you-a-more-productive-programmer/article-gif.webm", type: "video/webm", },]} poster="3-bucket://2018-04-28-why-prettier-will-make-you-a-more-productive-programmer/article-gif-poster.jpg" />

The super cool hero animation on https://prettier.io/

What is Prettier? Prettier can best be described as an "opinionated code formatter", it formats your code and that's it. It doesn't do all that ESLint does and it's not supposed to, the real benefit of Prettier is the set it and forget it configuration. There are a few options but for the most part, it's already done for you. I use Visual Studio code as my preferred editor when working with front end code and the Prettier plugin makes the process of code formatting instant, non-intrusive and almost invisible. Prettier can be set to format before commit or run manually but I prefer format on save. It makes it a seamless experience and it's nice to be able to have a cramped react component break into a well-formatted block when you start adding some more props. The great thing about Prettier is it can be integrated into existing workflows, it can even work alongside of ESLint.

How does Prettier improve my productivity? When something is opinionated, it cuts down decision fatigue. There is a right way to do it and no room for arguments. Prettier is no exception, what you sacrifice in customization, you make up for in time saved. You don't waste time to sifting through someone else code fixing up styling errors. You don't waste time reformatting a React component when you decide to pass down one too many props. You use your time to do what's important, get stuff done and the formatting is taken care of. The benefit of Prettier is not thinking about where there should be dangling commas or how long a line should be before it's broken up. It just happens and it's good at its job.

So what are you waiting for? If you're not using Prettier already, give it a go. Try it in your personal projects and see if you like it, talk to your team about it and spread the word. You'll discover how much time you save when some of the hassle is taken away.

If you aren't using a linter for JavaScript or you're looking for some inspiration, take a look at the Airbnb JavaScript style guide.

Thanks for taking the time to see why I think Prettier is a great tool for you're programming workflow. Let me know what you're experience is with JavaScript linters and any other tools you find helpful.