Gajus Kuizinas

Follow 6.8K Followers About

Stop using package-lock.json or yarn.lock

Gajus Kuizinas Sep 26, 2019 · 2 min read

I maintain over 200 repositories on GitHub and one of the most common PRs that I receive is someone adding `package-lock.json` or `yarn.lock`. These PRs are closed without merging because dependency lock files are not designed to be used by packages that are themselves dependencies of other packages.

What's going wrong?

Official NPM documentation encourages to commit `package-lock.json` files to the source code version control:

> It is highly recommended you commit the generated package lock to source control: this will allow anyone else on your team, your deployments, your CI/continuous integration, and anyone else who runs npm install in your package source to get the exact same dependency tree that you were developing on. Additionally, the diffs from these changes are human-readable and will inform you of any changes npm has made to your node_modules, so you can notice if any transitive dependencies were updated, hoisted, etc.

– https://docs.npmjs.com/files/package-locks#using-locked-packages

Committing `package-lock.json` to the source code version control means that the project maintainers and CI systems will use a specific version of dependencies that may or may not match those defined in `package.json`. Because `package-lock.json` cannot be added to NPM registry (by design; see NPM shrinkwrap), projects that depend on a project that uses `package-lock.json` will themselves use `package.json` to resolve

project's dependencies, i.e. what works for project maintainers/ CI systems might not work when the project is used as a dependency.

The origin of this misuse is NPM documentation. It should instead explain that `package-lock.json` should only be committed to the source code version control when the project cannot be a dependency of other projects, i.e. `package-lock.json` should only by committed to source code version control for top-level projects (programs consumed by the end user, not other programs).

I have already asked NPM to update the documentation, but it was archived without an action.

Responding to criticism

Some comments suggested that the biggest advantage of `package-lock.json` is that it allows to replicate development environment.

I would support a variation of `package-lock.json` if it could somehow only apply to `devDependencies`. I can see *some* (albeit small) benefit to wanting your development environment not break if there is a broken release. I would personally prefer my environment to break and become aware that a dependency in my toolkit requires attention (and depending on the nature of the issue either offer help, subscribe to an issue or replace the dependency). After all, you can easily patch your dependency tree if you need to lock down a specific version for development purposes.

However, there is no such option and using lock files at the moment will create the risks described in this article — namely that the dependencies that you use do not match those that your users will depend on. Responsible development requires that your script works with the latest versions of dependencies (and yes that includes transitive dependencies) satisfied by semver.

JavaScript Nodejs NPM