THE TIDELIFT MAINTAINER ADVANTAGE

How a lone maintainer of over 450 JavaScript packages keeps them well maintained and secure

Jordan Harband maintains over 450 npm packages, so when, during the summer of 2022, he was laid off for 18 months in a tough economy where few companies were making long-term growth investments, he was very grateful to have income coming in from Tidelift and its customers (Tidelift pays maintainers based on factors like customer usage and package criticality). The millions of developers who rely on Jordan's packages should also be grateful, because this income made it possible for him to continue his work to secure and properly maintain his projects without interruption.

Jordan lives in the Bay Area and has a family, so a dependable income is a must. And because he actually gets paid recurring monthly income by Tidelift and its customers for his maintenance work, he could be a full-time maintainer until he found a new job—the right new job, a job that he enjoys doing, and that lets him continue to dedicate time to working on his many, many open source projects.

"The fact that [I had income from Tidelift] allowed me to be a lot more calm and measured with the way I approached a job hunt," Jordan said. "I was able to take more time to look for the role that I wanted, instead of feeling like I was forced to take a role just to check the box and pay the bills."

The 2024 Tidelift state of the open source maintainer report found the top thing maintainers dislike about their work is that they are not financially compensated enough or at all for it. Number two is that they feel under-appreciated or like the work is thankless, and number three is that it adds to their personal stress. Thanks to income from Tidelift and its customers, Jordan was able to minimize the impact of these issues that have forced other maintainers to quit.

"During this 18 month layoff period, it would have been understandable if I had just dropped my activity on a lot of my open source projects because I was dealing with the stress of finding a new job, or because I needed to do different work that I didn't like as much in order to pay the bills," said Jordan. "So it sounds easy to underestimate the value of this and say that it's not quantifiable. But I don't think you can underestimate the benefit of having a drastically less stressful conversation with your partner about finances, and when you are laid off, that is of immeasurable value. I think that is one of the biggest concrete benefits I've gotten from Tidelift."

Thankfully, Jordan was eventually able to land a job where his employer understands and appreciates his important open source work, and allows him to work on projects like TC39 (the committee that writes the specification for JavaScript) as part of his job, while it also checks the box of allowing him to utilize his open source experience, expertise, connections, and perspectives.

"It's hard to measure how my free time efforts on open source would have been impacted by my soul being even slightly more crushed in my job," said Jordan.

Meanwhile, Tidelift customers using his packages probably never saw any outward signs that any of this was going on behind the scenes, because maintenance and security updates on his packages continued normally through that entire period.

A world without Tidelift: no more minimistjs?

In a world without Tidelift, Jordan says he would probably have had to take some sort of JavaScript engineer job, which wouldn't have been the end of the world, of course, because Jordan loves coding.

But it might have meant he couldn't continue to work on TC39, and it might have meant he wouldn't have had time to take over a project like minimistjs, a heavily depended-upon library that the previous maintainer had planned to delete because npm required two factor authentication and he didn't have a phone.

Jordan found out that the maintainer planned to delete his GitHub account when he was chatting with him via an X/Twitter DM. "I immediately had this moment of panic," Jordan said, "because some of the projects I maintain were under his username. But also, a lot of the packages I maintain depend on things that he owned."

Jordan offered to take over maintenance on all of the packages, and was able to get the minimistjs maintainer to transfer a few of them. But when it became clear that there was not an easy "one click" way to transfer ownership of the rest of the packages, the maintainer lost interest and went ahead and deleted his account.

Jordan recalled watching as the packages started to disappear, one by one, until they were all gone. After trying a few other things to see if he could recover or manually recreate them, Jordan reached out to GitHub to see if there was any way to undelete the packages and transfer ownership, and called in Tidelift to help make the case for him to take over ownership of the packages.

Luckily, since Jordan is already a Tidelift partnered maintainer, paid by Tidelift to implement enterprise class secure software development practices across many of the JavaScript packages he maintains, he was able to add minimist to the list of packages he receives income for from Tidelift.

And because this income is made possible by Tidelift customers, Jordan has already documented many of the important secure development practices followed by minimistjs, including providing vulnerability fixes for the latest release, monitoring dependencies for issues, having 2FA enabled, having a secure vulnerability disclosure process, and more. Even better, the attention from the recovery effort allowed a second maintainer to be onboarded, reducing the bus factor and maintenance burden.

The middleman of dependencies

Jordan's open source sweet spot is transitive dependencies. He lives in the glue work, the projects that hold the whole JavaScript ecosystem together.

"I didn't choose to be a middleman," Jordan said. "It's more that I really enjoy thinking about problems in a way that the Unix and JavaScript small modules philosophy encourages. I like the ideology and mindset and philosophy of, 'how can I take the problem to be solved, break it up into the smallest possible components, and then for each of those small problems, solve it so thoroughly, that I literally never need to touch it again.'"

One example of this is tape, a test framework for node and browsers that is downloaded almost 500 times a week. Jordan not only maintains tape, but also 86 of its 99 transitive dependencies. And because Jordan is a co-maintainer on nearly all of them, he can easily go in and fix any bugs that might show up. That's the beauty in maintaining so many projects, each a small piece of a big puzzle; if one piece breaks, just go in and quickly fix it, and the whole chain is fine.

"It continually baffles me why large organizations don't build things with this in mind, because why wouldn't you want the majority of your application to require no maintenance, almost no domain knowledge," Jordan said. "You could fire everyone and hire someone new, and they'd still be able to maintain the bulk of your application. Because it's so easy and requires so little effort, it just seems like the way everything should work to me."

Keeping the dependency chain intact

This way of keeping these dependencies maintained sometimes creates conflict for Jordan, though, because not everyone agrees with his approach. In fact, Jordan recently faced a GitHub repo controversy over replacing some dependencies in a project he was given commit access to with libraries he maintains.

"I think strong, separate modules, with more dependencies are better," Jordan said. "It's not overkill to make a separate one- or two-liner."

The left-pad debacle of 2016 is an example Jordan cites, when the maintainer unpublished all of his 200-plus modules from npm, which resulted in millions of broken builds and failed installations.

Many thought it was pointless to make a separate package for a functionality left-pad provided. However, because of Jordan's work on the padStart proposal for TC39, he had found bugs in every potential left-pad replacement he'd found online.

"But if you write an independent module, and you do it well," Jordan said, "you're gonna test all possible inputs to your function and therefore if someone uses it in a way you don't intend, you cover it. You can't misuse it."

Managing the demands of hundreds of projects at once

The other side of managing so many distinct projects is the backlog of work that needs to be done to keep them up to date and incorporate necessary features can be extremely large. For example, one of the projects Jordan maintains is called `resolve`, and provides node's file resolution logic for `require` in JavaScript. But it is not up to date with support for the `exports` field, which is related to ESM resolution, so there is a lot of module system complexity that needs to be added. Jordan has spent quite a bit of time figuring out how to do this work in a way that is manageable and secure, but he hasn't yet had time to bring it over the finish line.

Jordan also maintains 3 of the top 10 most used eslint plugins, yet it took almost a year to get to the point where two-thirds of them support eslint 9, and to ensure that they all support flat config, which is the new config format that eslint came out with in version 8, and became the default in version 9.

While Jordan is able to complete an incredible amount of work, given the scale of his maintenance efforts across hundreds of packages, there is not yet time for everything. That's where he gets most excited about seeing Tidelift's reach expand to more customers using his projects, so he can expand the capabilities of his projects as well, including in scenarios like these.

"I think Tidelift income has the potential to provide solutions to a lot of the problems I just listed. If sufficient income was to materialize, I would start a company, I would hire engineers, and I would entrust all of my projects to that company," said Jordan. "I would do that tomorrow if I had the cash to mobilize that effort."

As Tidelift recruits more and more customers who are users of Jordan's projects, those projects will continue to thrive and grow, and maybe Jordan will be able to start that company before too long. But even today, Tidelift customers can build their JavaScript applications with confidence, knowing that Jordan has made the commitment to ensure his packages follow enterprise secure software development practices. What's more, Tidelift customers directly played a role in funding Jordan's work on these packages so that Jordan can continue to make investments in ensuring they stay resilient and healthy into the future.