uShip takes JavaScript to new places with npm enterprise

uShip is the global transport marketplace that's reinventing the traditional logistics process.

By moving the logistics process entirely online and to mobile devices, uShip brings greater automation, transparency and efficiency to the traditionally inefficient shipping industry.

Number of Modules in npm Enterprise: 20

Development team: 35

Development Languages

.NET | JavaScript

The challenge

A .NET shop, uShip uses JavaScript for its front end. The company performs continuous integration, deploying code six to seven times per day through TeamCity. Prior to npm Enterprise, uShip faced a huge amount complexity when it checked dependencies into its Mercurial source control system. "We'd add one package, and we'd suddenly have 3,000 file changes," explains Bryson Reynolds, software developer at uShip. These enormous change sets ultimately slowed down upgrades to uShip's JavaScript tooling.

The solution

uShip found npm Enterprise to be a safe, reliable way to speed up its workflow for npm modules and prevent unmanaged dependencies. Although total build times have increased slightly since its packages are no longer in source control, the overall build process has improved.

"If Node is part of your continuous integration build pipeline, npm Enterprise gives you control over your environment."

—Daniel Poindexter

front-end architect

To learn more, visit www.npmjs.com/enterprise

Streamlined builds simplify day-to-day work and enable faster upgrades

npm Enterprise has sped up the workflow of adding npm packages to the frontend build system while minimizing the impact on deploy processes and everyday work. “The entire JavaScript ecosystem is moving quickly, so the ease of updates has helped us a lot,” says Daniel Poindexter, front-end architect at uShip. uShip has now set up npm Enterprise as a post-build step on its static project and has configured it to point to its local environment so only approved modules can get deployed. uShip is looking to extract the components that make up its static project into dependencies, taking common libraries and turning them into private modules hosted on npm Enterprise. “Pulling some of the dependencies out of the static project will help people that work with that code on an everyday basis,” Reynolds says

Breaking out code improves quality

Reynolds expects that the move to managing common libraries as private modules will also improve code quality. Reynolds continues, “It may seem easier to write one massive function that does everything, but your code gets better when you start breaking it into pieces with fewer responsibilities.” With npm Enterprise, these modular functions are easier to manage.

Confidence in JavaScript and continuous integration

Armed with npm Enterprise, Reynolds and his team have a much better way to incorporate npm modules in its continuous integration environment. “Just like you wouldn’t give everyone sudo privileges on a server, you don’t want people installing modules without any controls in place,” Reynolds concludes. “As more companies move to modular code development with continuous integration, they will appreciate the control and reliability of npm Enterprise.”

npm Enterprise has sped up the workflow of adding npm packages to the front-end build system while minimizing impact on uShip’s continuous integration process.

The ability to prevent unmanaged dependencies has increased confidence in using npm modules.

Upcoming work to manage common libraries as private modules will improve code quality.

“Most people don’t even notice npm Enterprise, and this is a good thing.”

—Bryson Reynolds

software developer

npm

To learn more, visit www.npmjs.com/enterprise