Nick Donfris

Dallas, TX · 817-723-6007 · donfris.nick@gmail.com · github.com/ndonfris · supplycode.dev

SUMMARY

Full-stack engineer (5+ years) specializing in developer tooling: language servers, editor integrations, and compiler-adjacent systems. Sole maintainer of fish-lsp, a production LSP implementation (~50k LOC TypeScript, thousands of weekly npm downloads) supporting Neovim, VS Code, Helix, Zed, and coc.nvim. Contributor to the Language Server Protocol specification and to tooling in several major IDEs. Background in compiler design (University of Arizona, under Dr. Ravi Sethi). Comfortable owning a project end-to-end — architecture, CI/CD, package distribution, and community support.

CORE SKILLS

- Languages: TypeScript (primary), Rust, Python, shell, Lua, C
- Domains: Language Server Protocol (LSP), tree-sitter grammars, parser/compiler internals, CLI & shell tooling, editor extension APIs
- Stack: Node.js, React, Next.js, Docker, GitHub Actions, npm/Verdaccio publishing, WebSockets, Rust
- Practices: property-based testing (fast-check/vitest), functional-first design in multi-paradigm languages, CI pipelines for cross-platform testing

EXPERIENCE

Maintainer & Author · fish-lsp (open source) 2021 – Present

- Designed and built a full LSP implementation for the Fish shell from scratch: completions, diagnostics, hover, go-to-definition, and semantic analysis over a tree-sitter grammar.
- Built a JSON Schema ecosystem for server configuration, with client-specific wrapper schemas for VS Code, Zed, Helix, coc.nvim, and Taplo.
- Implemented incremental/streaming completion handling (isIncomplete + timeout-based approximation) and typo-tolerant "did you mean" suggestions using Optimal String Alignment distance scoped by tree-sitter node context.
- Owns full CI/CD: GitHub Actions matrix testing (including Fish install + tree-sitter WASM builds) and dual-channel npm publishing (stable/nightly) validated through Verdaccio.
- Built an LSP stdio proxy/recorder for generating end-to-end test fixtures from real client sessions.
- Explored a browser/WASM build of the server (container2wasm, Web Worker transport) and a Fly.io-hosted WebSocket bridge powering a live in-browser playground with a custom CodeMirror 6 syntax highlighter.

Software Engineer · Independent Contractor 2019 – Present

- Delivered full-stack products across varied stacks for short-term clients as sole or lead engineer: Next.js apps with SSR/SSG, React Native mobile apps, and Three.js/WebGL integrations for interactive 3D experiences.
- Comfortable ramping quickly across unfamiliar codebases and tech stacks - a recurring requirement of project-based work.

AI Model Evaluation Contractor · Outlier · Alignerr 2022-Present

- Evaluated and improved training data for frontier AI labs, including structured review of AI-generated code for correctness, subtle defect patterns, and adherence to best practices.
- Built real-time, WebSocket-based browser tooling for live evaluation of model input/output, used internally to support training workflows.

OPEN SOURCE & STANDARDS

- Contributor, Language Server Protocol specification (microsoft/language-server-protocol).
- Contributed patches and integrations to multiple major IDE/editor ecosystems (Neovim, VS Code, Helix, Zed).
- Maintain a suite of Fish shell tooling (tmux/fzf integrations, abbreviation generators, completion utilities) used as reference implementations by other Fish users.

EDUCATION

B.S. Computer Science · University of Arizona 2017 – 2022

- Wildcat Excellence Scholar · Arizona Excellence Scholar
- Studied Compiler Design under Dr. Ravi Sethi (co-author, "Dragon Book")