Will Ruggiano

[GitHub] [LinkedIn] [Discord: willruggiano]
[Email: crowd.must.cue@usecloaked.com]

My background is actually in architecture (i.e. undergrad) and I have been mostly self-taught in the ways of coding (though I have had several incredible colleagues and mentors along the way). It started with Java – I wrote my first main method with the help of a Dummies Guide, while on holiday with my family in Mexico. Years later I learned C++ while working on a real-time streaming protocol at AWS, and this really sparked my interest in the lower levels: the network stack, memory management, the *type system* and compiler. Around that time I also found Nix and Neovim which has since fueled many a late night coding session. I currently work at a startup, and we are currently pivoting after a failed first iteration. As part of this new discovery phase (of the startup and my own personal adventure) I've been learning a new language, Zig, and becoming increasingly interested in machine learning. I've primarily fueled this interest by (a) reading academic papers (see below), and (b) applied learning via things like Karpathy's Zero to Hero lectures.

Experience

Software Engineer, Tendrel

2023–

- The usual web stack: React, Typescript, NextJS/GraphQL, Postgres.
- Led the development and release of our most "successful" application.[1]

Software Engineer, Amazon/AWS

2016–2023

- Joined Amazon out of college.
- Spent four-ish years on the retail side with the Notifications team. More or less: SOA, Java, DynamoDB.
- Worked with two phenomenal engineers to design, build and deploy a new notifications service, and then migrate/deprecate the legacy (Perl) service.[2]
- Stumbled upon Apple's iOS rich-content push notification APIs, and then led its integration into Amazon's iOS mobile application.[3]
- Transitioned to AWS to try something new. Landed on a team building a real-time streaming protocol for AWS's VDI products.
- Worked primarily with one other engineer on the core protocol/shared library. C++(17), UDP, STUN/TURN and a little bit of Python mixed in for sanity.

Undergrad (Architecture), University of Virginia

2012–2016

- I loved Legos as a kid, so architecture seemed like the perfect fit :D
- First coding experience: Python in Grasshopper (Rhino 3D).

[1]This was the only application to see organic sign ups (hence "successful") among the handful of apps that we attempted to launch.
[2]This had two big effects: (1) During migration, I uncovered a bug in the legacy system that resulted in a substantial amount of high volume notifications being dropped (~20%), which meant a lot of money down the drain. I got a promotion out of it :) (2) We cut our infrastructure bill by 70% (even with the 20% bump in traffic).
[3]I don't have raw numbers on this one but we targeted the high volume use cases first (enabled by above[2])… so, somewhere on the order of the count of Amazon retail orders.

Code (and friends)

tendrelhq/graphql

There's some interesting stuff in here, mainly around the so-called "state machine" mechanism. This repo was the backend for the Runtime application during my time at Tendrel. A rather detailed description of how this application was intended to work can be found here.

neovim.drv

This is my personal developer environment, packaged as a Nix flake. You'll find mostly Lua and Nix code in here, all for the configuration of neovim. For yet more Nix code see the library I use to do the heavy lifting.

nix-community/neovim-nightly-overlay

I am a co-maintainer of this repo. Its purpose is to make nightly neovim accessible to the Nix ecosystem. Unsurprisingly you'll find only Nix code in there.

dotfiles

I guess I'll include this one since it demonstrates a modicum of knowledge about Linux and "system administration". It's mostly Nix code (do you see a pattern?) and contains the system configurations for my laptop and desktop.

Friends (i.e. skills)

I am quite capable of learning new languages and skills on the job, in fact I would say that this is one of my greatest strengths! Examples of this include learning Typescript (not that impressive) and C++ (much more impressive) as part of my day job at Tendrel and AWS, respectively. In both cases I became a subject matter expert for my team/org/company.

	Versions	Proficient?	Last used	Example
AWS		yes	2025	
C++	11, 14, 17	recently	2023	
Docker[4]		not not	2025	tendrelhq/graphql
Java	8	once upon a time	2019	
Linux	5+	enough to be dangerous	curr.	dotfiles
Lua	luajit/5	yes	curr.	neovim.drv
Nix	2	yes	curr.	neovim.nix, vscode-js-debug.nix
Postresql	14+	yes	2025	pg_jsonpatch[5]
Python	3	recently	curr.	
Shell	bash, fish, zsh	yes	curr.	
Typescript	5	yes	2025	tendrelhq/graphql
Zig	nightly	learning	curr.	

[4]Works great with Nix :)

[5]Fun example of Nix making compat testing a piece of cake

Reading

[1] S. Mimram and C. Di Giusto, "A Categorical Theory of Patches," *Electronic Notes in Theoretical Computer Science*, vol. 298, pp. 283–307, 2013, doi: 10.1016/j.entcs.2013.09.018.

[2] A. Vaswani *et al.*, "Attention Is All You Need." [Online]. Available: https://arxiv.org/abs/1706.03762

[3] M. Kleppmann and H. Howard, "Byzantine Eventual Consistency and the Fundamental Limits of Peer-to-Peer Databases." [Online]. Available: https://arxiv.org/abs/2012.00472

[4] A. Auvolat, D. Frey, M. Raynal, and F. Taïani, "Byzantine-Tolerant Causal Broadcast," *Theoretical Computer Science*, vol. 885, pp. 55–68, 2021, doi: 10.1016/j.tcs.2021.06.021.

[5] C. Qian *et al.*, "ChatDev: Communicative Agents for Software Development." [Online]. Available: https://arxiv.org/abs/2307.07924

[6] S. Sarkar, D. Staratzis, Z. Zhu, and M. Athanassoulis, "Constructing and analyzing the LSM compaction design space," *Proc. VLDB Endow.*, vol. 14, no. 11, pp. 2216–2229, 2021, doi: 10.14778/3476249.3476274.

[7] J. Dormans, "Engineering Emergence: Applied Theory for Game Design." [Online]. Available: https://eprints.illc.uva.nl/id/eprint/2118/1/DS-2012-12.text.pdf

[8] I. Mirzadeh, K. Alizadeh, H. Shahrokhi, O. Tuzel, S. Bengio, and M. Farajtabar, "GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models." [Online]. Available: https://arxiv.org/abs/2410.05229

[9] Y. Lafont, "Interaction Combinators," *Information and Computation*, vol. 137, no. 1, pp. 69–101, 1997, doi: 10.1006/inco.1997.2643.

[10] J. Yallop and L. White, "Lightweight higher-kinded polymorphism." [Online]. Available: https://www.cl.cam.ac.uk/~jdy22/papers/lightweight-higher-kinded-polymorphism.pdf

[11] A. Meyer, "Range-Based Set Reconciliation." [Online]. Available: https://arxiv.org/abs/2212.13567

[12] A. Ananthaswamy, *Why Machines Learn: The Elegant Math Behind Modern AI.* Dutton (Penguin Random House), 2024.

[13] M. Cemri *et al.*, "Why Do Multi-Agent LLM Systems Fail?." [Online]. Available: https://arxiv.org/abs/2503.13657

[14] mikolalysenko, "Peer-to-Peer Ordered Search Indexes." [Online]. Available: https://0fps.net/2020/12/19/peer-to-peer-ordered-search-indexes/

[15] G. Wang *et al.*, "Hierarchical Reasoning Model." [Online]. Available: https://arxiv.org/abs/2506.21734

[16] T. Parr, G. Pezzulo, and K. J. Friston, *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior.* The MIT Press, 2022. doi: 10.7551/mitpress/12441.001.0001.