Devin Brite

New York, NY, USA
dwbrite@gmail.com
+1 (774) 293 8465

dwbrite.com
keybase.io/hd
github.com/dwbrite

I'm a software developer specializing in backend development and devops. I'm always seeking new opportunities to expand my knowledge and skills, especially in areas that are outside of my comfort zone.

Some other domains I've explored recently are embedded firmware, rendering, and electrical engineering. I'm particularly interested in pursuing projects that involve using physical objects with interactive elements, especially as they relate to the ambiance and *feel* of a space.

Experience

DevOps Engineer
June 2022 - Dec 2022

Remediant, Inc. - San Francisco, California (Remote / NY)

At Remediant I managed our SaaS infrastructure in the form of Kubernetes clusters and AWS services. I used Terraform to administer VLANs, deploy and configure our application and logging stacks, and implement monitoring and alerting systems to maintain SOC2 compliance.

Following a major outage, the company realized that it was almost impossible to redeploy our infrastructure from the existing codebase. To address this, I led the project to refactor our codebase, allowing it to be deployed from scratch for the first time. I focused primarily on breaking the codebase down into isolated modules with a clear dependency chain.

Full Stack Development Intern
Jan 2019 - July 2019

ProGlove - Munich, Germany

At ProGlove I was a core developer of ProGlove Insight - a cloud-based platform for IoT device management built on AWS. Insight enables analysis of scans from ProGlove devices, leading to further efficiency improvements in cu stomers' manufacturing and logistics processes.

As one of only two programmers who worked on Insight from the start, I played a key role in the design and implementation of the product, as well as in the interviewing and subsequent hiring of four additional team members.

Neat Projects and Explorations

Graphics Programming with WGPU

While I was experimenting with my custom game engine for a 2D game called void, I wanted to figure out a.) how to build my render graph, and b.) how to effectively render thousands of *unique* sprites.

In an attempt to resolve those questions, I made a particle rendering demo which draws and updates hundreds of thousands of individual particles every frame. Learning to make a renderer like this involved understanding low-level graphics APIs, writing compute, vertex, and fragment shaders in WGSL (and previously GLSL), and asking for *plenty* of help from niche online communities.

Infrastructure and Services on dwbrite.com

My website acts as a showcase of my skills and a platform for further exploration in backend development and infrastructure.

In the past, I've hosted it on a simple VPS using my own orchestration tool, dorc, which integrates with SystemD to create and multiplex green-blue deployments. Today I use Kubernetes on Linode to host dwbrite.com and other services that allow me to divorce myself from proprietary software and cloud-specific solutions.

Skills and Technology

Languages

All of my recent personal projects have been written primarily in **Rust**, because I feel it allows me to write code quickly using abstractions that tend not to bite. I started learning Rust in 2017 and haven't looked back since. I've even contributed to the Rust by Example book and written blog posts about my favorite features. But fear not - I'm not a zealot, nor a one-trick pony!

I have a few years of **Java** under my belt (from the Java 8-9 era), I've created a Pokemon clone(*) in **Kotlin**, I've written APIs professionally in **Go** (which I'd like to work more with), and I have a handful of smaller projects in various languages including Python, C++, CoffeeScript, and JavaScript.

Databases, Caches, and Storage

I learned **MySQL** in high school to create an anarchistic “for students, by students” newspaper website.

The nth iteration of my *personal* website used **PostgreSQL** to store blog posts. My current website stores blog posts with the source code in git, but uses Linode's **S3**-compatible object storage for storing media like gifs, videos, images, and binaries.

I've also used **DynamoDB** as a cache for timely access of recent time-series data.

And More!

I've been using **git** consistently since 2017. Since turning git into guitar hero by attempting to rebase within the Jetbrains GUI, I've become *well*-acquainted with the CLI. I have a strong preference towards linear commit histories.

My operating system of choice since is generally Fedora or a Debian derivative. I dual-boot **Windows** on my desktop and make an effort to ensure my projects compile on both platforms.

I worked with **AWS** at ProGlove and Remediant, and I'm familiar with a large portion of its services - notable ones being EKS, VPC, EC2/ALB, Lambda, and Cloudformation. More often, I tend to use **Linode** for my personal projects.

I'm pretty quick to learn, so don't be afraid to ask for technologies that aren't listed here. I'm particularly interested in learning GitOps, as well as Scala, Ansible, and Istio.