resume Documentation

Release 1.0.0

Chris McDonald

Jul 08, 2019

Contents

Hi there, my legal name is Chris McDonald but I prefer Wraithan. This is my web resume. I'm looking for a job, but I'm only interested in Portland, OR based jobs or jobs where I could be remote from here.

Who is he?

- A polyglot who enjoys the commonalities and contrasts of each programming language and platform.
- A native of Portland, OR, USA. (And is not looking to relocate)
- Loves tuning systems for performance and stability.
- A nerd, cyclist, and metal head.

I am a Node.js and Rust hacker. Though I have years of Python development experience as well. I enjoy enabling more stable performant software in any way I can.

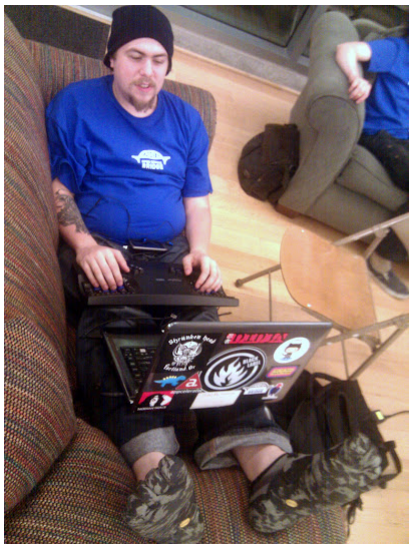

Fig. 1: Open Source Bridge 2011.

In my spare time, I teach programming, play with hardware, and hack on side projects. I subscribe to a statement I heard in one of the keynotes at PyCon 2012: "When programming stops being fun, I'll stop doing it."

CHAPTER 1

Table of Contents

1.1 About Me

I'm a software developer from Portland, OR. I spend most of my programming time in Node.js and Rust. I spent a significant chunk of my past working in Python, so much so I have a large tattoo of one on my left shoulder! I enjoy exploring programming languages and chatting about them.

1.1.1 Motivators

Making code as bullet proof as possible. I've found few things in software give me the same sense of satisfaction as knowing it is extremely hard to crash my code. Rigorously going through code and making sure all error cases are accounted for and writing tests to validate it is how I can comfortably sleep at night with my code running on critical servers.

Next is probably making code more performant. Whether this is algorithm work, caching, query tuning, or any other optimization work. Profiling something, changing some code, and seeing it get faster always puts a smile on my face.

Putting those together in some eloquent solution is what I always strive for in my projects. I just enjoy cleaning up technical debt. Usually this means going into a system that as evolved for a while and needs to be distilled into clean concepts again. Taking a messy system and bringing it to order is delightful.

Finally I love teaching folks, whether it is a 1:1 situation or a meetup full of people. When I was just getting started in software, the folks who took the time to teach me had a huge impact on where I ended up today. I am always trying to pay it back to the community, whether I'm organizing meetups, mentoring friends or helping folks out online.

1.2 Technical Skills

This is a list of my stronger technical skills. I've played with writing things ranging from graphics engines, to decoding game save file formats, to window managers, to IRC bots, to web sites.

Language Skills:

- Deep understanding of Node.js.
- Have worked with Python, C++, and SQL in the recent past.
- Have played with Rust quite a bit.
- Have some work experience in the further past in PHP, Perl, and Java
- Others I've played with in some capacity or another: Lua, Haskell, Common Lisp, Clojure, and Go

1.2.1 Node.js

It sparked the fire in me to really start enjoying JavaScript. I have used it to build out micro services, command line utilities, libraries, and for work.

- Have a deep understanding of performance in Node.js and V8.
- Wrote instrumentation for a number of database drivers and frameworks, requiring knowledge of the library internals.
- Wrote a benchmarking suite as a series of micro services that can run benchmarking and load testing jobs for days or weeks.
- Participated in various contributor discussions affecting the direction of Node.js for tracing and release cycle.
- Have played in lots of online AI games using JS.

1.2.2 Rust

This programming language is easily my favorite at the moment. The ownership system and the type system both feel like a great advancement over other modern languages. It has really reawakened my desire to work at the native layer instead of in VMs.

- Primary language I use on the weekends for learning and small projects.
- Spent time reimplementing past Node.js AI bots of mine in Rust and seeing the differences in algorithms.
- Language of choice for code challenges like Advent of Code
- Past few years of personal game development have been using this language.

1.2.3 Python

I've spent about 6 years programming in Python, 4 of that was professional. Most of my experience with Python is centered around writing Django applications.

Some things I've built outside of web applications:

- A command line task management system
- Screen scrapers
- Feed aggregators
- Plugins and extensions for various tools that embed Python.

Django

A couple years ago If I was going to write a web app, I would probably start with `django-admin.py startproject <project name>`. I used it for years on projects both large and small.

Here are some highlights of from Django projects I've worked on.

- Three sites that shared the same code base and served a large number of users.
- OAuth2 (spec 10 and 11) based signup and authentication.
- Upgrading between multiple Django versions.
- Numerous community and small business sites.

Celery

I use this when I need to offload tasks in Django based sites. Here are a few things I have done with it:

- Helped architech and develop a lazy caching backend that updated itself out of band using Celery, or calculated in line if celery hadn't updated the cache yet.
- Divided tasks into separated queues so the Celery daemon could be shared to multiple servers.

Fabric

This tool has saved me hours, if not days of my life.

- I have ran 2 sprints on it, one PSF sponsored, the other at PyCon.
- Made deployment simple and very reproducible causing it to be fast and take care of all the repetitive details for the team.

1.2.4 Redis

I reach for this when I want a key/value store or centralized pub/sub. I have use it for:

- Django caching backend.
- Django session storage.
- Celery queue backend (including connection pooling)
- Micro services based IRC bot using Redis' pub/sub as a transport.
- Chat system for a video game

1.2.5 PostgreSQL

This is my preferred relational database. It scales pretty well, it is open source, and I've come to rely on it anytime I need a database.

- Have used tools such as pgfouine to profile and optimize usage.
- Used pgbouncer to do connection pooling to decrease latency.
- Have scaled to tables with millions of rows.

1.2.6 MySQL

Used this in a pretty large database (150+ tables) which required a deeper understanding of relational algebra than I had previously. Learned many of the quirks around time handling and strict mode SQL as I wrote queries to gather user and game state from the database.

1.2.7 Git

I am commonly found teaching people how to use git, recover from situations they and not sure how to get out of, and giving my opinions on best practices based on experience and discussion with others that have passion about how to use their version control system.

1.2.8 C++

Working on Dropzone at Sparkypants meant developing async C++ libraries for use by the game client and game server to communicate to the backend services.

In working with hardware I've had to relearn and get better at C++. It was my first language, so coming back to it after spending years doing other development is quite a bit of fun. Most of the development has been for arduino compatible chips, communicating with the outside world using serial.

I also have had to read a lot of C++ while inspecting the internals of Node.js and V8, developing my ability to read other people's C++ in complex environments.

1.2.9 BigQuery

Telemetry data at Cedexis/Citrix was shipped to this database, most of my work with this database was finding was to effectively query many GB to TB of data and get out statistically relevant insights out.

1.3 Other Skills

A listing of my less technical skills and development methodologies that I wasn't sure where else to put.

1.3.1 Agile

I have never worked anywhere that implemented any of the agile methodologies fully. But I have worked with several of the techniques found in Scrum and XP, including:

- Pair programming
- Continuous integration
- Sprint based development
- Test driven development
- Collecting and writing user stories

1.3.2 Public Speaking

Public speaking, especially educating a group on a topic is a lot of fun to me. Here is a list of places I've spoken or ran sessions

- I was the co-organizer of PDX Node for a couple years.
  - I spoke on a number of topics from library introductions, to code refactoring, to hardware hacking.
  - Ran the hack night and hardware hacking sessions.
  - Helped put together and mentor at multiple NodeSchool sessions.
- I gave my first conference talk at NodePDX. It was on using Redis Pub/Sub in node.js.
- PDX Python (Most recent to oldest)
  - Building Distributed Systems with Redis and Pub/Sub
  - django-slow-log
  - inspect (lightning talk)
  - dis (lightning talk)
  - baker (lightning talk)
  - Using bpython
- Barcamp Portland 2012
  - Co-ran a session on building distributed systems had people writing services for ZenIRCBot in the session to show how easy it is.

1.4 Projects

The vast majority of my projects are open source and can be found on GitHub. This is a list of projects I've written or contributed to in some way. I love discussing them, so feel free to ask me about them.

1.4.1 Shader Playground

shader-playground is a learning project of mine, the code isn't the prettiest but has given me a lot of understanding of the shader pipeline and graphics programming in rust. Also, this project I spent time playing with fractals like Mandelbrot and Buddahbrots.

- Created a bunch of art, learning about how to generate and save it in various formats.
- Gained a deeper understanding of complex numbers and various ways to think about them.
- Was able to transfer this knowledge to other game development projects I have in progress, making them more technically solid as well as convenient features like screenshotting and debug output.

1.4.2 Message Broker

Message Broker was a project where I attempted to write an enterprise worthy message broker akin to RabbitMQ, Kafka, or Redis' Pub/Sub layer. I planned on open sourcing once I got it to a usable state, but work and other side projects took precedence.

- Building on top of Tokio which is becoming a community standard for building services, protocols, and as a layer to manage async machinery.

- I'm also writing a best practices book to go along with it, as an exercise in that type of writing. This way I design the message broker in a usable way that encourages good system architecture.

1.4.3 md-to-wp

md-to-wp is a small rust project I coded up in an afternoon to convert markdown to a WordPress friendly HTML. This includes colorizing my code examples. You can see it in action this blog post.

1.4.4 i-can-manage-it

i-can-manage-it is a game and game engine I started writing to get used to the world of game development. I haven't open sourced it yet, but I did do some writing about it as I learned and encountered problems.

- First Rust project where performance really mattered, so I got to spend some time optimizing.
- Had to learn and relearn linear algebra in order to do the coordinate transforms to do camera effects and object manipulation I wanted.
- Wrote a debug socket server that I hooked up to an Electron project that displayed stats about my running game engine.

1.4.5 cargo-bump

cargo-bump is a command that increases the version number of the Rust project you are currently in. It is meant to be a Rust version of npm version.

- A fun exercise in gluing libraries together in Rust.
- An interesting opportunity to work with others in the community who are writing Rust libraries.
- Started as a challenge to myself to build a Rust project from scratch in a night.

1.4.6 WeeChat Notifier

weechat-notifier is a daemon written in Rust meant to connect to a running WeeChat session on another machine and provide notifications on the local machine. I am writing it mostly as an experiment in writing a parser, client library, and a daemon in Rust.

- Learned a valuable lesson about how writing a parser in a static language like you would in a dynamic language results in cumbersome code.
- Had a lot of fun experimenting with different testing styles.

1.4.7 Read the Docs

Read the Docs is a site for building and hosting sphinx documentation. The main goal of it is to lower the barrier to writing docs as much as possible. The idea is that if there is free hosting, automated building, and easy to select themes developers would write docs. Once they are written maintaining them is easy because when you push your docs are automatically rebuilt.

What I did for the project:

- Core developer/maintainer on the project for a couple years.
- Wrote better integration with GitHub, including tests.

- Made it possible for multiple people be admins on a project.
- Took part in architecture discussions with the maintainer.
- Took over maintainership for 4 months while the previous maintainer was away.

1.4.8 ZenIRCBot

ZenIRCBot is a IRC bot that works a bit differently than your standard bot. Features (and interesting to implement things) include:

- Microservice architecture
- Redis pub/sub as a transport
- Services can be written in any language.
- Core bot written in Node.js but reference implementations are also in Python and Clojure.
- Patched various third party libraries to enable features in each of the bots.

1.4.9 PDXNode

This is less a software project and more a project that revolves around, software. I was a co-organizer for the group for years, which meant planning meetings, getting speakers, running workshops, and more.

- Once a month software hack night that I mentored at.
- Once a month hardware hack night which I was a core mentor for.
- Once a month presentation night, some months I filled in if we couldn't find a speaker.
- Periodic workshops, some I was the main organizer, others I was just a mentor.

1.4.10 Hardware

Since July or so of 2013, I've become a hardware hacker to complement my software development skills. It is really great being able to interact with the real world, not just via a keyboard or mouse or something.

- Built a monitoring system for my house, temp, light, humidity and cat door.
- Built a media keyboard (play/pause/back/forward/volume) where I had to patch the firmware that comes with the chip to better adhere to the USB HID spec.
- In the process of building a bike computer with GPS, heart rate, cadence and a light system.
- In the process of building a monitoring system for my smoker.

1.4.11 Resume

That would be this. I have it up on GitHub because it is easier to maintain there. It is written in ReStructured Text using sphinx so I can host it on Read the Docs.

I was inspired by another user who has his resume on Read The Docs as well!

1.5 Gigs

My professional work in the software industry over the years. There is a bit of time between jobs here and there where I was working non-technical jobs and spending a lot of time on personal *projects* working to develop the skills needed to work in the industry.

1.5.1 Cedexis / Citrix

Senior Engineer (Oct '17 - Current)

With a background doing telemetry in JavaScript from working at New Relic, I took an opportunity to work on a browser based RUM client and related infrastructure. What made this job particularly interesting to me was having telemetry data used for more than just alerting or performance analysis, it was fed into intelligent DNS server that used it to make routing decisions.

- Redesigned and built Dynamic Content benchmarks to include a cryptographic signature, giving us the first verified measurement type, ensuring no caching happened that would invalidate the measurement.
- Built out the canary deployment system for the RUM client, allowing us to roll our releases or experiments out in 3-5% increments. This enabled us to try out changes to benchmarks and fixes for particular browsers without risking our whole corpus of data.
- Provided deep analysis of benchmarks before and after major changes like shifting from HTTP to HTTPS or changing which browser API is used to gather the data. Giving us confidence and ability to notify customers if they'll see a step function in their data.

1.5.2 Sparkypants Studios

Senior Backend Enginer (Feb '16 - Jun '17)

I got an opportunity to work on a video game, which had always been a dream of mine. As backend engineers we built the systems that stored data, matchmaking, login, and provided C++ libraries for the client and game server engineers to use. We used a microservice architecture written in Node.js, Python, SQL, and C++. Heavily using RabbitMQ as our message bus and MySQL as our database.

- Redesigned the matchmaking backend to enable a confirm step before creating the game, while refactoring all data collection from the database to be just in time instead of up front eliminating a large class of bugs and AFK users.
- Built an automated deployment system for our staging environment, with fully automatic registration and decommissioning of game servers on top of systemd and ansible.
- Built and designed a full consent friend system to systemically prevent abuse where possible.
- Refactored a C++ wrapper around libcurl to standardize header parsing and error propagation, preventing accidentally ignored errors and making HTTP access code easier to write.

1.5.3 New Relic

Node.js Agent Engineer (Apr '14 - Feb '16)

My first full time gig working in Node.js, and it was an amazing experience. I was able to connect with so much of the Node.js community via my work for New Relic. The Node.js Agent is the library New Relic customers install in their Node.js applications which monitors it and sends data back to New Relic servers for analysis.

- Maintained a weekly release cadence for over 52 weeks in a row. Allowing the agent to respond quickly to the needs of our users.
- Ran a series of support training session giving our support engineers much deeper understanding of how Node and our agent works. Resulting is a large reduction in support escalations.
- Learned about the deep internals of Node.js and v8 in order to make our agent more accurate and performant.
- Wrote and designed systems that accounted for as many error cases as possible because an error in the agent could crash a user's system.

1.5.4 Contracting

Web Developer (Oct '13 - Apr '14)

I took on a few small contracts in this time, mostly though I spent my time learning to be a better Node.js engineer as I wanted to take my career toward more asynchronous programming.

1.5.5 Mozilla

Web Developer (May '12 - Sept '13)

This was my first time working in truly high scale development, both in traffic and in team size. It was also my first gig working with a purely remote team distributed across many time zones.

- Started out working on addons.mozilla.org, reworking the use of Redis as a part of the caching solution.
- Worked on security critical sections of the site including the blocklist that Firefox uses to shut down bad addons and extensions in the wild.
- Was moved over to working on marketplace.firefox.com as part of the payments team.
- Integrated with multiple payment providers and built the security pin portion of the site.
- Was involved in many of the architecture choices such as revamping deployment, moving to smaller services, and caching.

1.5.6 Aquameta

Senior Software Developer (Mar '10 - Feb '12)

I loved this company and learned a great deal while I was working there. It is where I cut my teeth on Django apps that needed more than just some more hardware thrown at them to scale.

- Was part of a team that implemented, maintained, and extended a large Django application that powered 3 sites.
- Scaled that application using celery for offloading and generous amounts of caching.
- Upgraded Django between major versions twice. From 1.1 to 1.2 and from 1.2 to 1.3.
- Wrote and encouraged the writing of both unit tests and functional tests.
- Wrote and maintained one click deployment scripts using Fabric.
- Interfaced with the clients regularly to gather requirements for features.
- Guided the architecture of the application using community best practices and past experience.

1.5.7 Parthenon Software

Software Developer (Sept '09 - Nov '09)

This was a PHP shop I worked at for a short while. I did development on a well established code base.

- Updated unit tests, allowing for more confidence that application was correct.
- Met with clients to discuss and advise on what course to take for re-designing their software.
- Implemented feature requests, fixing existing bugs in the module while adding the feature, resulting in cleaner, better documented code.
- Participated in "brainstorming" sessions concerning design/testing details for various project.

1.5.8 Critical Path Software

QA Tester (May '08 - Aug '08)

Here I worked in the QA department testing software and hardware. The primary project I was hired for were 2 lines of computers that a company was going to release and they wanted some independent stress testing done in a wide range of activities. Online gaming, word processing, downloading content, watching HD video both streaming and off a Blu-Ray.

- Learned to write very effective bug reports.
- Wrote and executed test plans, tracking progress and reporting defects.
- Worked with a team to decide on software milestones and requirements.
- Set up many different hardware/software configurations for testing.
- Wrote a tool using C++ to generate data for testing.
- Assist in delegation of various portions of testing to help train new members of the team prior to product release.

1.5.9 Transim Technology

Intern Software Developer (Dec '05 - Aug '06)

This was my first foray into the world of software development at a company. The stack was a large java backend with a PHP layer on top with liberal use of Perl as glue.

- Cleaned up and maintained several in-house tools written in Perl, Java, and PHP for processing and displaying circuit schematics.
- Created a GUI for two of the in-house tools so that non-technical staff could assist in processing schematics that needed human interaction.
- Implemented a secure login system with detailed permission setup.
- Documented all of the above mentioned work, along with a large portion of a Java based webserver back-end.

I had a great time at this job and this, on top of my passion I already had, really sealed the deal as far as my desire to pursue software development as my career.

1.6 Contact

So, I've intrigued you enough that you want to contact me.

- email: xwraithanx@gmail.com
- twitter: @wraithan

If you want my phone number or other contact information, use one of the above methods and talk to me about it. There are obvious reasons I'm not going to put my address or phone number on the 'net.