Resume of Glenn T. Barry

Table of Contents

1 Description

Seeking a software development position in Atlanta (or remote) utilizing my skills and job experience detailed below.

2 URLs for this Resume

2.1 Web site: https://gaak99.github.io/resume

2.2 Raw formats including plain text: https://github.com/gaak99/gb-res

3 Contact

3.1 email: gaak99@gmail.com

4 Computing Technologies

4.1 Github

4.2 Languages

4.2.1 Bash

1. *autogen.sh* fix for Remacs project github pull request
2. *oxmerge* is a shell wrapper for the *oxly* common case github
3. see also Emory below

4.2.2 C

1. see Sun Microsystems below

4.2.3 Lisp

1. Clojure
    1. Concurrent recursive "grep" using *Clojure.core.async* Go-style channels github
2. Basic bloom filter implementation in Emacs Lisp github

   I used the dash uber modern/functional Lisp list library.

4.2.4 Python

1. *oxly* auto-merges Dropbox file revisions with a *git*-like CLI github

The side project that I put the most time into and use on a regular basis.

2. see also Seagate below

4.2.5 Rust[1]

1. Remacs open source project - "oxidize" (port) C layer of GNU Emacs to Rust github

   I ~~ported~~ re-wrote several low level directory/file functions from C to Rust.

   I contributed 98% (as of Aug '18) of the code in the dired Rust files.

   I used some functional style Rust features here in dired unix.rs (see function *fattrs to list* (replace each *spc* with *underscore*)).

4.3 Operating Systems

4.3.1 Unix

1. Ubuntu Linux

   My dev laptop runs Ubuntu 16.04. See also Seagate below.

2. Solaris / OpenSolaris

   See Sun Microsystems below.

3. SmartOS

   This cloud optimized derivative of OpenSolaris has some unique features for cloud devops such as DTrace.

4.4 Web Frameworks

4.4.1 Flask

1. see Seagate below

4.4.2 Django

1. see MeetMeTix below

4.5 Cloud Services

4.5.1 AWS S3 API

1. see Seagate below

4.5.2 Dropbox API

1. *oxly* side project github

4.5.3 FaunaDB[2] Service API

1. *ATL100* side project github
2. *hylisp* side project github

4.5.4 Openstack Swift object storage API

1. see Seagate below

4.5.5 rsync.net

1. ZFS as a Service

   It's one of my favorite backup services as not only does it provide a familiar Unix file system interface (via sshfs) but also provides read-only snapshots several times a day.

5 Job Experience

5.1 June 2013 - June 2015, Seagate/Evault, San Francisco, CA.

I was one of the first engineers hired to this project that started as a skunkworks within Evault – a wholly-owned subsidiary of Seagate – and last half of my tenure there we were brought into Seagate for the new Cloud Storage division.

It was run like a startup using Agile methodology and Jira/Confluence/Stash (git server).

5.1.1 Software engineer, Cloud Storage R&D Group.

1. The product
   1. LTS2.evault.com - cheaper and faster retrieval (vs AWS Glacier) of long-term object storage

      The service was based on Openstack Swift[3] object storage open source project.

      Swift is many thousands of lines of Python 2 and is heavily object oriented. And the code tends to be of high quality as it has many eyes of Python experts for code review.

Swift runs on a Ubuntu cluster (we had one cluster in Salt Lake City and later added a second one in Oklahoma City).

All cluster inter-node messaging was RESTful and gave me a good (sometimes intense) experience debugging these type services including down to the Python *WSGI* module.

A teammate wrote a custom user/customer DB that the other services accessed using SQLAlchemy.

My projects:

1. I "owned" Swift AWS S3 emulation module aka fixed many bugs
2. Data center to data center object replication performance improvements
3. Basic web site features (password reset, etc) with Python/Django
4. Internal microservice APIs with Python/Flask

5.2 Sep 2012 - Jan 2013, MeetMeTix.com, Atlanta, GA.

5.2.1 Contract programmer, Python/Django Web Apps.

1. Major Projects:
    1. Single sign-on for the site

       Integrated/tested/refined Django-Social-Auth app for Facebook single sign-on feature.

    2. Developed (Python) from scratch a full Selenium automated test suite.

5.3 Oct 2001 - Jun 2012, Sun Microsystems (acquired by Oracle in 2010), Menlo Park, CA.

5.3.1 Software engineer, Solaris OS/Net Security Technologies.

1. Last few projects:
    1. Solaris secure boot feature

       Refined initial design with an emphasis on next generation SPARC systems.

    2. Diagnosability improvements of Solaris Secure NFS/SMB (C development)
    3. Kerberos KDC server db replication (C development)

       I led team and co-designed/developed RPC-based DB replication protocol/feature for the Kerberos KDC.

4. Solaris kernel RPC GSS modules server performance improvements (C development)
5. Solaris single sign-on via GSS/Kerberos (C development)

5.4 Oct 1996 - Oct 2001, Sun Microsystems.

5.4.1 Software engineer, Solaris Sustaining OS/Net Name Services.

1. Diagnosed/coded/tested fixes (C language) for escalated bugs.

5.5 June 1987 - Sep 1996, Emory University MathCS Dept, Atlanta GA.

Unix Systems Administrator.

6 Education

June 1981 - 1987, Georgia Tech, BS in Information and Computer Science.

Footnotes:

[1] Wikipedia

> Rust is a systems programming language sponsored by Mozilla which describes it as a "safe, concurrent, practical language", supporting functional and imperative-procedural paradigms. Rust is syntactically similar to C++[according to whom?], but its designers intend it to provide better memory safety while still maintaining performance.

[2] Infoq

> FaunaDB is a transactional, temporal, geographically distributed, strongly consistent, secure, multi-tenant, QoS-managed operational database. It's implemented on the JVM for portability, and it's relational, but not SQL. Instead, it's queried via type-safe embedded DSLs, like LINQ. FaunaDB is a return to the general database purpose model, but built for the cloud instead of the mainframes of the 80s.

[3] Openstack Wiki

> The OpenStack Object Store project, known as Swift, offers cloud storage software so that you can store and retrieve lots of data with a simple API. It's built for scale and optimized for durability, availability, and concurrency across the entire data set. Swift is ideal for storing unstructured data that can grow without bound.

Author: GT Barry
Created: 2018-11-09 Fri 16:00
Validate