Bradon Kanyid

bradon@kanyid.org • www.kanyid.org • github.com/rattboi

Core Technical Skills

Proficient Languages: Python, Groovy, Go, C, C++, Bash, Assembly (ARM, x86, Z80, PIC, 68k), Verilog
Familiar Languages: Ruby, Clojure, Java, .NET (VB, C#), LaTeX, SQL, Rust
Software: Platform Agnostic (Linux/BSD/Mac/Windows), Embedded GNU Tools (GCC, GDB, Redboot), VCS (Git, Svn, IBM RTC), Gradle, Ant, Xilinx ISE WebPACK
Hardware: Digital Design, PCB Layout, FPGAs, SMD Soldering

Experience

- **Software Engineer** — **UTi Worldwide Inc.**
  2013 – Current — *Portland, OR*
  - Introduced Gradle technology as a migration path away from legacy Ant build system. Wrote a templated multi-level orchestration engine for managing TIBCO BusinessEvents technology stack. Developed an automatic deployment program focusing on service-level orchestration. Created TDD-based Groovy library for build/deploy tasks. Wrote deployment monitoring tools to centralize deploy reporting across eight separate prod and non-prod environments. Implemented custom add-ons to support ChatOps automation to further centralize monitoring and automation of disparate systems. Working to implement CI/CD via Docker and Jenkins.
- **Automation Engineer** — **Silver Bay Seafoods, LLC.**
  2009 – 2013 — *Craig, AK*
  - Wrote ladder logic for automating plant's sensors and actuators, such as conveyor belts, hydraulic rams, joysticks, and heat-sealers. Developed touchscreen Human Machine Interfaces and SCADA for monitoring and controlling the automation systems, data-collection middleware between automation systems and business software using .NET and SQL. Developed internal company website for remote observation and statistics in ASP.net.
- **Tech Support Representative & Internal Technician** — **POS-X Inc.**
  2007 – 2009 — *Bellingham, WA*
  - Support for POS-X products, managing trouble tickets, e-mail, phone support, hardware repairs, and mass computer assembly. Developed network-based automated burn-in and imaging system for new computers using PXElinux, BartPE, and Norton Ghost.

Major Projects

- **Auto Deploy (Groovy)** — **(internal tool at UTi)**
  I completely rewrote the legacy UTi Deploy frontend tool. It uses a similar but extended specification language, and supports many new features including build artifact validation, deploy ordering, parallel deploys, simple dependency management, and simultaneous multiple deploy targets. Created Groovy-based decoupled, reusable, testable components in a shared Build and Deploy code library for future projects. This library includes a Spock test suite, Cobertura instrumentation for code coverage analysis, CodeNarc code quality static analysis, and SonarQube continuous inspection.
- **Build Watcher (Go)** — **github.com/rattboi/build-watcher**
  To centralize the visibility of the build and deploy process at UTi, I wrote a log-watching program that forwards intelligent build and deploy results to a notification system (Slack) that summarizes the work in realtime. This is the first in a suite of tools to create a ChatOps system at UTi.

Major Projects continued on Next Page

- **GMusic-Local-Sync (Python)** **https://github.com/rattboi/gmusic-local-sync**
  This hobby program was written to help me sync my missing library of music to Google Music's cloud. Google has their own tool that works on audio fingerprint, but it only works on individual tracks. My music collection is album-based, so I wanted all-or-nothing import of entire albums. If Google already contains my album, use Google's version; if not, upload my local version. Determines if matches exist based on a set of heuristics, including album/artist similarity (Levinshtein distance) and filtering of extraneous keywords such as Expanded Release, Bonus Tracks, etc.

- **Mopidy-Subsonic (Python)** **github.com/rattboi/mopidy-subsonic**
  Mopidy is a music framework that decouples music service frontends and backends, e.g., it allows the use of Spotify and Youtube with a custom web frontend. I wrote an extension that allows the use of the music service Subsonic as a backend within the Mopidy framework. I maintain the Mopidy-Subsonic package on PyPI.

- **Linux Kernel Driver (C)** **github.com/rattboi/blec_dev**
  In Linux Device Drivers, I wrote a Linux kernel driver in C for a USB external input/output device. It supported stable hot-plugging and removal of simultaneous devices, with separate interfaces to each device.

- **Video Game Console Emulator (C / ARM Assembly)** **sourceforge.net/projects/wonderboi**
  Initially ported, then extended a PC-based emulator for a portable game console to another portable embedded platform. The final version of the emulator was almost entirely written by me. Wrote screen blitting/scaling, file i/o, graphics caching, UI, sound, memory mapping, and more.

Education

- **B.S. Computer Engineering** **GPA: 3.85**
  Portland State University *Magna Cum Laude, June 2013*

Honors Societies & Volunteering

- **Eta Kappa Nu** **web.cecs.pdx.edu/~eta/**
  IEEE Honors Society. Limited to top 25% of Department.

- **Womprats Audio Synthesizer** **github.com/killerfriend/womprats**
  Honorable Mention in Industry Design Practices course.

- **IEEE Student Store** **ieee.pdx.edu**
  Volunteering includes 4 hours per week of desk duties, as well as occasional weekend store resupply.

- **Computer Action Team (CAT)** **cat.pdx.edu/thecat.html**
  The CAT is a voluntary IT program for Portland State University's School of Engineering. Volunteer 4 hours weekly at the CAT front desk, helping students with computer and networking issues, as well as handling trouble tickets and maintaining student-related computer services. Test and develop new systems and services used by students and fellow CAT members.

- **International Aerial Robotic Competition** **devel.avt.cecs.pdx.edu/projects/iarc**
  Participated in developing an autonomous quadcopter (4-rotor helicopter). Contributed to integrating various sensors to the quadcopter, specifically an infrared range finder for flight height information. Wrote software for embedded LPCxpresso flight computer. Developed documentation for other parts of the low and high-level systems.

References Available upon Request