Bradon Kanyid

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

Core Technical Skills

Proficient Languages: Python, Groovy, Go, C, Bash, Clojure, C++, Assembly (ARM, x86, Z80, PIC, 68k)
Familiar Languages: Javascript, Java, SQL, Ruby, CoffeeScript, Rust, Kotlin
Software: Terraform, Packer, Docker, Ansible, Puppet, Gradle, ANT, Jenkins, XL Deploy, Git

Experience

- **Build & Release Engineer** — **Standard Insurance**
  2017 – Current — *Portland, OR*
  Developed declarative GitHub auditing/policy config management tool to manage Standard Insurance GitHub organizations. Created Jenkins pipeline library components to support Sonarqube continuous quality, Qualys security scans, provisioning ephemeral test environments using Terraform, and Selenium-based integration tests. Developed Gradle plugins to manage Terraform/Packer/JBoss-related builds and deploys. Lead pilot project for moving multiple projects into AWS. Implemented Nagios monitoring for test environments and build/release infrastructure. Increased developer visibility of builds/deploys/monitoring via Slack notification routing engine. Developed Ansible roles for in-house infrastructure and deployment of various Java-based technologies.

- **Software Engineer** — **IBM**
  2016 – 2017 — *Portland, OR*
  Aided in decoupling OpenStack's CI system (Zuul) from the OpenStack infrastructure to make it generally available as a GitHub integration. Assisted in development of Ansible-based continuous deployment architecture to manage a system of OpenStack virtual machines. Developed open-source chatbot and web interface to manage asynchronous standups for our globally-distributed team.

- **Software Engineer** — **UTi Worldwide Inc.**
  2013 – 2016 — *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.

Education

- **M.S. Computer Science** — **GPA: 4.00**
  Portland State University — *Started 2018 / In Progress*

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

Major Projects

- **Conifer Routing (Python)** **(internal tool at Standard)**
  To increase visibility of our developers products during build/deploy process, I created a solution that interfaces between Nagios, Jenkins, and other tools, and various messaging tools (primarily Slack and e-mail). Conifer is a Flask application that allows developers to create rules for routing messaging events to their preferred notification system. This allows teams to subscribe to build failure events, or service failures in a dev environment, for example.
- **Github Audit (Groovy)** **(internal tool at Standard)**
  At Standard Insurance, my team was in charge of new git repo creation within managed organizations. This included many policies around our specific branching model, branch protections, team access, and webhooks. To help manage this, I developed a tool that used the Github API to determine which repositories did not fit our current policy, and generated events to determine the best way to rectify any issues found. Some policy violations, such as incorrect repo access, were automatically rectified by the tool.
- **BonnyCI (Python)** **bonnyci.org/beta/**
  BonnyCI is an in-progress open-source continuous integration product based on OpenStack's Zuul. Zuul is highly coupled to the OpenStack project and its Gerrit code review frontend, so there were many changes necessary to decouple these technologies and generalize to be usable as a GitHub integration. BonnyCI is meant for large-scale projects that have complex issues including cross-rep dependencies, and automatic gating of code commits until other dependent commits are merged.
- **Hoist (Ansible)** **github.com/BonnyCI/hoist/**
  Hoist is a repository of Ansible roles to manage the BonnyCI project. Bootstraps BonnyCI on a host of environments, including OpenStack, Vagrant, and Docker. After initializing, self-manages the continuous deployment of any changes to the Hoist repository. I added many features to Hoist, including integration with ARA (Ansible Runtime Analysis) to enable faster debugging.
- **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.
- **GMusic-Local-Sync (Python)** **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.
- **Video Game Console Emulator (C / ARM Assembly)** **github.com/rattboi/wonderboi32/**
  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.

References Available upon Request