Bryan Bugyi

Software Engineer
https://linkedin.com/in/bryan-bugyi

June 22, 2020
bryanbugyi34@gmail.com
(609)500-7081

Summary

I am a Software Engineer with a very "systems"-oriented skill set. In particular, I have the following areas of expertise / specialization: **Python** programming, **Linux**, and **Networking** (TCP/IP)

Education

- **Rutgers University** — New Brunswick, NJ
  B.S. Computer Science and Mathematics — *2015 - 2019*

Industry Experience

- **Edgestream Partners, L.P.** — Princeton, NJ
  Software Engineer — *May 2019 - Current*

  As a Software Engineer on the Production team at Edgestream, my work centered around the improvement / maintenance of Edgestream's production trading system. This system is Linux-based (Red Hat) and is written primarily in Python. A few notable achievements associated with my work at Edgestream are listed below:

  - Integrated pylint into a codebase with well over a million lines of existing code. As the lead on this project, I was responsible for building several non-trivial tools to aid in making this integration a success.
  - Made several large-scale improvements to the production department's in-house testing framework/runner.
  - Lead developer of Edgestream's investor-facing web portal, which was built using Python's Django web framework.

- **Comcast** — Mount Laurel, NJ
  Tier III Technical Support — *2011 - 2016*

  Tier III Technical Support at Comcast was a liaison group that worked to resolved chronic customer issues as well as outages which were impacting multiple customers. This involved troubleshooting Comcast's network to resolve the issue directly or, failing at that, escalating to the appropriate Engineering group and working with them to diagnose and resolve the issue cooperatively. Some of the other responsibilities associated with this role are listed below:

  - Monitored for large-scale network outages by aggregating and analyzing customer issue data from our ticketing system.
  - Used Python scripting to automate frequent department tasks (e.g. parsing spreadsheets, opening/closing tickets, information retrieval).

Skills

- **Languages:**
  I have assigned each of the languages listed below a rank between 1-4 to indicate my level of proficiency with that language. A rank of 1 indicates that I have taken a college course that made use of the language and/or have read a book about the language. A rank of 2 indicates that I have also written over 5,000 lines of code in the language. A rank of 3 indicates that I have also written over 20,000 lines of code in the language. And a rank of 4 indicates that I have also written over 50,000 lines of code in the language.

  Programming Languages

  - Python: 4
  - C/C++: 3
  - Bash: 3
  - Perl: 2
  - Rust: 2
  - Haskell: 1
  - Java: 1
  - Javascript: 1

- **Frameworks:** Django, Flask, Pandas, Twisted
- **Technologies:** Docker, Git, PostgreSQL, RedHat, Vim

Open Source (GitHub) Projects

- **psf/black** Python
  The uncompromising Python code formatter. *2019 - 2020*
  - Improved the way black handles strings (https://github.com/psf/black/pull/1132).
  - Non-trivial contribution consisting of ∼3,500 lines of code additions/modifications.
  - Led to the closing of five unrelated GitHub issues (opened by five different developers at different times).

- **bbugyi200/cookie** Shell
  A Template-based File Generator. *2018 - 2019*
  - Well received by the developer community (over 200 stars on GitHub).
  - Multiple outside contributions have been accepted (e.g. user submitted issues/bug reports and code contributions).

- **bbugyi200/funky** Python, Shell
  Makes shell functions easier to define, more flexible, and more interactive. *2017 - 2019*
  - Well received by the developer community (over 400 stars on GitHub).
  - Multiple outside contributions have been accepted (e.g. user submitted issues/bug reports and code contributions).

- **GermainZ/weechat-vimode** Python
  A WeeChat script that adds vi-like modes, commands and keybindings. *2017 - 2018*
  - Made several non-trivial contributions to this project. My most significant contribution was a parser that I wrote which was used to replicate vim's `:nmap` command (∼500 lines of code).