Bill Auger

Musician, Student, Teaching Assistant, Freelancer
Active candidate interested in full-time, consulting, internship.
Open to working remotely. Willing to relocate.

45 West Main Street, Dudley, MA 01571
http://bill-auger.github.io
bill-auger@email.com

About me:

Avid musician (guitar, percussion, keyboard, etc)
Student of Computer Science, Logic, Philosophy, and Music
Volunteer teaching assistant for the CS169 ruby-on-rails online course on edX
Freelance and open-source software developer

Experience:

Education

- Background in music, electronics, and design
- Technology and software enthusiast
- MOOC-aholic

Freelance Software Developer

Technologies:
LSL, PHP, RubyOnRails, C#

Description:
About five years ago, I shelved my musician hat and began writing codes for online business owners in virtual-worlds such as SecondLife; including 3D object scripting (LSL), custom bot clients (C#), database back-ends (PHP/Ruby) for those with large amounts of data requiring persistence and/or websites for those desiring a web presence. Recently, I have also found a niche contributing codes and documentation to free-and-open-source projects through the tip4commit service.

Responsibilities:
Typical responsibilities include: consulting with clients to determine requirements, estimating the project time-frame, deriving and formulating a specification, coding, acceptance testing, and delivering/installing the final product.

Free and Open Source Software Developer

Technologies:
Smalltalk, C, C++, RubyOnRails, Javascript, PHP

Description:
Check out some of my pet (read: portfolio) projects listed on the next page for a sampling of my interests and abilities.

Responsibilities:
These are mostly tools that I have made for my own use and to hone my skills in various languages, interesting methodologies, and useful libraries. One of them however (Bridgin), is used daily by hundreds of students and teaching assistants for the online RoR course on edX for which it was written.

Projects:

TeamStream **(C++)** **Author** **2012 April - 2012 May**

A NINJAM client specialized for team-streaming. Based on the original NINJAM Win32 source.

https://github.com/bill-auger/teamstream

Loopidity **(C++)** **Author** **2012 August - 2013 December**

A multitrack looping audio recorder for GNU/Linux designed for live handsfree use. Requires the JACK and SDL libraries.

https://github.com/bill-auger/loopidity

Webjam **(C++ / Javascript)** **Author** **2013 March**

A NINJAM client with javascript frontend. Based on the original NINJAM ncurses source.

https://github.com/bill-auger/webjam

Bridgin-PHP **(PHP)** **Author** **2013 August**

A nifty php-cli script for bridging pidgin conversations (superseded by bridgin).

https://github.com/bill-auger/bridgin-php

Bridgin **(C)** **Author** **2013 September - 2014 April**

A purple plugin for bridging pidgin conversations.

https://github.com/bill-auger/bridgin

Chuck Song Builder **(ChucK)** **Author** **2013 October**

Some chuck classes for notating ditties.

https://github.com/bill-auger/chuck-song-builder

Saas-Puppy **(ISO)** **Author** **2014 April**

A minimal RubyOnRails development environment for students based on PuppyLinux.

http://puppyisos.org/saas-puppy.html

LinJam **(C+)** **Author** **2014 May - 2014 July**

A cross platform NINJAM client with linear chain-streaming mode.

https://github.com/linjam/linjam

About the musical me:

Bill performs what he refers to as an “eclectic ‘un’-set" of classic rock, blues, country, jazz, and traditional standards in his own unique "funkacoustic" interpretations; and is well known in the online live music community for taking requests. A genre unto himself, somewhere between Pat Boone and Black Sabbath; his "style" has been described as "Stevie Wonder meets James Taylor in the parking lot of a Pink Floyd concert".

His musical influences include Jim Croce, Bob Dylan, Jerry Garcia, Jeff Lynne, Roger Waters, Neil Young, Frank Zappa, at least 3 Beatles, and up to 4 Willburys (to name just a few); but his typical sets will equally butcher several genres.

About the nerdy me:

I got my first computer (a TI994A) as a young pup which may be some indication of my age (read: wisdom). When I was a schoolboy, I'm fairly certain that there was not a teacher within 50 miles who could find the power switch on a computer (they were usually in the back or underneath - true story) so I taught myself to program TI-Basic; which is as I figure the best way to learn anything. I fondly remember the day when I told my math teacher about a computer program that I wrote to do my homework for me; which caused him to become noticeably perplexed as to whether this was deserving of extra credit or disciplinary action.

Around the age of 20 I took a 2000 hour certificate course in Electronics / Computer Repair (covering the gamut from chemistry to machine code); which gave me the background in hardware that proved to be invaluable once I began learning the C language.

Once the wonderful world web came around, I immediately took an interest in HTML and Javascript. Then around 2003, I introduced myself into the wilderness of GNU/Linux; so bash scripting, system administration, and networks were my next endeavors (quite serendipitously really – on the count of my Windows broke). These remained mostly hobbies for me until recently though, born of curiosity and the desire to have an adaptable and resilient computing environment.

Around 2008, I decided to develop my long latent interest in programming more fully and began familiarizing myself with other languages and paradigms (Smalltalk, PHP, C#, Ruby, and LSL) and more recently thanks to edX and Coursera (Python, SML, Racket, ChucK, Oz, Prolog, Eiffel, and Haskell).

Of the languages that I've mentioned above, I would say that by far the ones that to me were the most influential were Smalltalk and Racket. When I first started reading about Smalltalk something 'clicked' with me and all of the mystery and intimidation of programming dissolved at once. Self-consistent and liberated from rigorous syntax; this was the language that I wished I had been able to learn first. One of the books that I read at the time was "The Joy of Smalltalk"; which at first, struck me as a bit of an overstatement; but within a few months, I found that title to be quite accurate. Finally, I had found a truly expressive language that seemed more musical than mathematical. Instead of being purely logical, it seemed sensible; and rather than a rigorous tool you had to 'learn' by rote rudiment, it was a nice, fuzzy, flexible toy that you could 'understand' intimately through creative experimentation. As I've heard it said: "Instead of writing code to make the machine happy, I could write code that made me happy.". That, more than anything I think, fueled my new found passion in programming and my eventual gravitation toward Ruby, Rails, and the Agile Methodology. Racket, I would like to give some 'props' as well for being a 'sensible' and self-consistent language in the functional paradigm; which was a welcome change after previously studying SML. It was in learning Racket that I became comfortable in the functional paradigm and that I now actually like Javascript; now that I am no longer working it "against the grains", as they say. The functional programming class I was taking did not allude to this, but it was another of those ”a-hah” moments for me the day I realized “Hey, Javascript is actually just the ugly duckling of functional languages.”.

In summary, I would say that if one is familiar with the similarities and differences between Java vs. Smalltalk and between an ML vs. a Lisp; and one can appreciate the elegance of Smalltalk and Racket over their more obtuse counterparts; then much could be inferred "about me" and my general philosophy regarding programming and style. A propensity for “rocking out” is also helpful.