Alex Beal

alexlbeal@gmail.com
www.usrsb.in

Education

University of Colorado at Boulder **2007-Present**

- BS Computer Science – In progress
  - Track: Software Engineering
- BA Philosophy – Completed 2010 with Distinction
  - Interests: Ethics, particularly Virtue Ethics
- GPA: 3.82

Phi Beta Kappa Honor Society **Inducted April 2010**

- Chapter: Alpha of Colorado at the University of Colorado

Projects

Program Analysis for Reflection and Strings **Present**

- **Description**: I am currently involved in an undergraduate research project with Professor Evan Chang, and Devin Coughlin, a PhD candidate in the CS department. We are interested in how reflection is being used in dynamic languages, such as Python and Ruby. For my part, I am writing a dynamic analysis for Python programs. This program takes an arbitrary Python program as input, runs the program, and outputs information about the program in real time. Some of the data includes: when strings are created, when they're operated on, and when they are used in reflective calls. Eventually, we will be able to track the flow of all strings through an arbitrary program. The source code isn't posted publicly yet, but I'd be happy to give a demo in person.

Python to x86 Compiler **Fall 2011**

- **Description**: For a compiler construction class, I pair programmed a Python to x86 assembly compiler with my classmate, Josh Wepman. The compiler accepts a Turing complete subset of Python, which includes classes, objects, if-statements, and arithmetic operators, and outputs a text file containing x86 assembly. This can then be assembled into an executable using an assembler like GCC. For the end of the semester project, we developed a tail call detection analysis, and extended the compiler to optimize tail calls.
- **Report**: http://is.gd/tail_call_report
- **Github**: https://github.com/beala/bullfrog

Blog - /usr/sbin **2011-Present**

- **Description**: I maintain a blog at www.usrsb.in where I write about whatever programming and mathematical topics I happened to find interesting. I love to share my knowledge, and also find that writing about a topic helps me to understand it myself. Here are some highlights:
  - **Building Data Structures from Functions**: How to build linked lists from computation's most basic building block—the lowly function. http://is.gd/data_from_functions
  - **Picking Random Items From a File:** An exercise in algorithms, where I develop a memory efficient method for picking random elements from a file. http://is.gd/random_items

  - **A Zero-Knowledge File Exchange:** A proposal for a file exchange website where the operators of the site cannot access the files being exchanged. It uses the principles of the Diffie-Hellman key exchange. http://is.gd/zero_knowledge_exchange
  - **Super Bowl Probabilities: The Coin Toss**: A fun post where I examine just how improbable it is that the NFC won 14 coin tosses in a row. http://is.gd/sb_coin_toss

WORK EXPERIENCE

Archiiv, Inc. **May 2008-November 2008**

- **Position**: Web Developer, Systems Administrator
- **Description**: Archiiv was designed to be a photo sharing website tailored specifically to the needs of historical societies. As a web developer, I used the following languages and products: PHP, MySQL, JavaScript, Google Maps API, BASH Scripting, and HTML. As the systems administrator, I also maintained the Linux servers (Slackware) that were used for web development.

TECHNICAL SKILLS

Programming Languages

- **Python**: This is my go to language for projects large or small. It's elegant, flexible, and lends itself to high productivity. It's by no means a silver bullet, but is flexible enough for many tasks.
  - Favorite parts: Dynamic typing, generators, an elegant syntax, its built in data types, and its commitment to the principle of least restriction (i.e., it lets you do whatever you want, which can be both dangerous and fun).
  - Least favorite parts: Its lack of tail call optimization and reluctance to embrace the functional programming paradigm.
- **C**: This is the first language I ever learned, and now I've been using it extensively for an OS class I'm currently taking.
  - Favorite parts: The core of the language is very small. It offers a lot of low-level control, and speed, but is still portable.
  - Least favorite parts: Its strict adherence to the imperative style, and its lack of higher level abstractions.
- **Scala**: I'm new to this language, but I have been using a subset of it for a programming languages class I'm currently taking. I'm amazed by how easily things like interpreters and type checkers can be expressed in it.
  - Favorite Parts: Integration of OOP and functional features, pattern matching, type inference, and preference for immutability over mutability.
  - Least Favorite Parts: Size of the language, speed of the JVM and compiler, and lack of tail call optimization.
- **Others**: I have a strong interest in programming languages, so I've brushed up against many others, including Scheme, BASH, PHP, Ruby, Java, C++, x86 ASM, JavaScript, and HTML. Some of these I haven't used in quite a while, but I could become productive in them without too much overhead.

Operating Systems

- I am familiar with all the major operating systems, and am very comfortable on the OS X and Linux command line. I spend almost all of my development time there.