Eric Zhou

zehric

ericfzhou@berkeley.edu

(805) 832-7323

zehric.github.io

SKILLS

node.js

java

python

sql

scheme

lua

C++

bash

unix

REST

HTML

CSS

EDUCATION

August 2015 - Present
University of California Berkeley
Electrical Engineering and Computer Science, GPA 3.78

Relevant Coursework:
CS61B (Data Structures)
EE16 (Designing Information Devices and Systems)
CS70 (Discrete Mathematics)
EE105 (Microelectronic Devices and Circuits)
CS188 (Artificial Intelligence)

EXPERIENCE

Summer 2016
Software Intern
Rently
Developed a Node.js natural language processor that allows users to control Rently Keyless devices with their voice. It takes natural language from the user, which is processed by an arbitrary service, then parses the natural text into JSON, which is then passed to the Rently servers to control a specific device.

PROJECTS

Text Editor
Course Project
A fully functional text editor in Java. Features include automatic word wrap, open and save, vertical scrolling, changing font size, undo/redo, and more. In order to implement constant time insertion/deletion from anywhere in the document, I used a data structure that combines the constant time access of array lists and the insertion/deletion of linked lists.

Bear Maps
Course Project
A web mapping application of the Berkeley area, using the OpenStreetMap project. Features routing that uses the A* algorithm to find the shortest path between two locations on the map. Implements map rastering with a quadtree of images of increasing resolution to support high quality display at any zoom level.

AnimeCal
Personal Project
A Japanese television animation calendar desktop application. It is written in Node.js and uses the Electron framework. Pulls information from the *AniList API*, organizes it based on air time, and displays it to the user with the front-end desktop application. I have plans to also make a mobile app for this.

SIXT33N
Course Project
Final project of the EE16 course series. It is a mobile robot on 3 wheels (2 drivable) that moves around according to speech input. It uses the MSP430 Launchpad as its guts with some circuitry for driving the motor and sensing through a microphone. Voice recognition is implemented with PCA classification and straight driving with eigenvalue placement in a feedback loop.