# Project Progress Report

Application for Interactive LED Visualization

Hayden Gomes
CSE 237D

## Background

The University of California, San Diego's (UCSD) Computer Science and Engineering (CSE) department building was constructed in 2005. The architects who designed this state-of-the-art teaching and research facility designed the building with a unique look and feel. Along these lines, they designed the floor plan of the building to be shaped like a foot. In the "big toe" of the building, the architects included tri-color LED fixtures. These iColor Cove fixtures, usually found in theater lighting systems and clubs, are the first model in a successful line of lights by Color Kinetics. With seventy fixtures lining the large windows on each floor of the big toe, these LEDs are able to fill the room with colorful ambient light.

Each fixture can be independently set to one of 16.7 million additive RGB colors through DMX-512 signaling protocol. However, since the building was completed access to these fixtures has been limited to a panel on each floor that can execute up to eight preprogrammed light shows. Also, because the big toe rooms have restricted access and the LEDs are most observable at night, there is little opportunity for students to use them.

During the fall of 2009 I worked with Catherine Wah and Emmett McQuinn on a group project for our compilers class (CSE231). The project goals were to develop a simple language for programming light shows and the software for compiling the language and displaying the results in a simulator or on the LED fixtures in the big toe. The project concluded with a successful demonstration of programs developed in our language being compiled and displayed on one floor of the big toe.

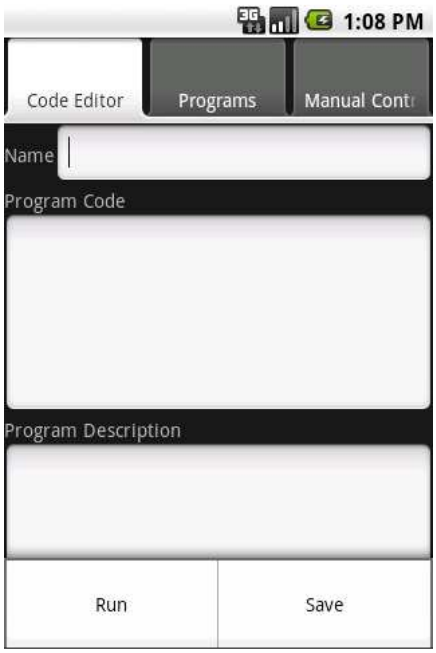
## Project Goal

The goal of this project is to develop a method for students to remotely control the big toe LEDs, increasing their overall accessibility and mindshare. This will be done by building on the compilers project from the fall quarter. I will create an application for Google's Android OS that will allow users to write programs in our language and then send them to a web server that will compile and execute the programs, displaying them on the big toe fixtures. Additionally, if this goal is completed in a timely fashion, the application will be expanded to include a method of manually controlling the fixtures and a simulator to allow users to view a visualization of their program before submitting it to the server.

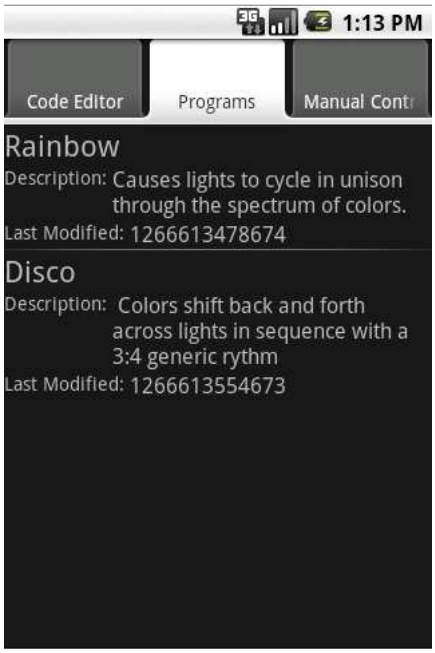
## Approach

This project has two parts: the user interface and server side execution engine. For the user interface, Google's Android mobile phone OS will serve as the base platform. The application will be developed on a HTC Dream (T-Mobile G1) phone. While the latest version of the Android OS is 2.1, this model only supports the Android 1.6 platform. However there is sufficient backwards compatibility between versions that the application should function on all Android phones.

The application will include a basic text editor for writing code, a database to store programs, and an execution option. The editor, as seen in the figure below, will allow users to develop their programs and provide a name and a brief description of what the expected visualization will be. In the future there is potential for a central database where all programmers can upload their code, in which case the name and description fields will be a necessity. After writing a program users can chose to save or run it. For storing the programs, a SQLite database will be used as it is already built into Android. A second programs tab will allow users to browse through a list of programs they have saved (see below). Through a context menu for the items in this list, users can load programs into the editor in order to adjust or execute the programs, or they can delete unwanted programs from the database.




Program Editor Tab




Tab Program List Tab

When the run option is selected for a program, the code is uploaded to the server where it can be compiled and executed. The majority of people using Android phones have data plans for them, and the phones also have built in WiFi. With internet access readily available, the code will be uploaded to the server through a web form.

For the server side execution engine, a basic apache web server will be used. As the original code from the compilers project was all written in python, the server websites will be written in Python CGI for development convenience. Once a program is received the code is passed into the compiler which performs basic syntax checks and then produces a four dimensional array of the visualization. This array is then transmitted through an Enttec USB Pro serial port using DMX-512 signaling protocol. The signal passes through the Color Kinetics power supply and out to the LED fixtures. Each fixture has a configurable address that it will use to decode the correct color channel values for its position. This signaling method allows for a refresh rate of approximately 30 Hz.

## Progress

A basic fully functioning version of the android application has been completed. Using the application I have successfully posted the contents of the editor to a test web form. The editor is more or less complete. As seen in the approach section figure. The saving and loading features are functioning correctly. I have tracked down a server in the 3rd floor big toe that I can use for hosting the web server and am currently getting an account for it. This server location is ideal as a wire needs to be run from it directly to the LED power supply. Also, working with a server that is already in place has the advantages of not having to track down a physical place to store it.

While waiting for access to the final web server, Catherine Wah has been helping me setup a temporary web server on another computer for testing purposes. We were able to create a web form that would take in code submissions and display them on the lights in the 4th floor big toe classroom where we did our original testing last quarter. The main issue that needs to be taken care of is that the server must be reset between visualizations.

## Future Work

The Android application could be polished off a bit. Through testing the current version I have found it to be somewhat difficult to type out code on the phone. I will try to come up with ways of making this easier for the end users such as providing example programs that they can build off of.

I have included a tab for the manual controls as shown in the approach section figures. The server side of manual control should be fairly easy as it will only need a second web form created that bypasses the compiler and goes straight to the display. However, the Android side interface is proving to be difficult. I would prefer if it could provide full manual control of all 70 fixtures on the floor, but I haven't been able to think of a clean and intuitive way to display it on the three inch screen clearly.

Another addition that I hope to make to the application is including a "how to" help section on programming the LEDs. Currently only the three members of my original team have written

code in our language and we haven't created any formal documentation on the syntax. This will make it difficult for other students to start using the application. So this help section will be important for the people running this application to be able to understand how the language works.

Emmett McQuinn is starting to look into ways of saving the simulator results so that they can be posted on a site for users to be able to visualize what their programs look like without needing to run them on the actual lights. If he can get this to work in time I will add an option on the Android to submit to simulation or to the LEDs and then try to display his simulator results on the Android.

On the server side, I need to work with Catherine on debugging the server to eliminate the need for it to reset after every execution. I will also need to finish up the designs of the web forms that the Android will be submitting to. Once I get access to the machine in the big toe I will transfer the code over to it and, with Dave Wargo's help, connect it to the fixtures.

A couple of features that will be necessary for the server before the Android application or website are publicly distributed include some form of limitations or override controls so that students aren't running the lights while people are trying to teach or work in the big toe. Also, ideally the site would be hosted on the UCSD-Protected network without being accessible from the internet in general.