John Sy
(808) 202-4395 | sy.john.r@gmail.com

Projects

3D Music Visualization with SoundCloud API

A primarily **JavaScript** web application served on **Flask** that randomly generates a 3D world with trees, fireflies, clouds, and a waterfall. A custom music player built for the **SoundCloud API** lets you choose your favorite track and watch as the fireflies move and change shape/color in response to your music.

- Uses **asynchronous** requests to the **SoundCloud API** to load buffered audio data from the resulting response into a custom music player featuring a **neumorphic design**.
- Utilizes **Howler.js** library and the **WebAudio API** to create an **audio context graph** and an audio analysis node to generate an array of integers that correspond to frequency data from your song playing in real time.
- Said frequency data is then fed into a **Three.js** scene which uses 3D terrain meshes created from geometric primitives that are placed randomly throughout the allowed 3D space.
- The frequency data informs a set of randomly shaped and placed fireflies whose **RGB** values, **vertical/y-axis displacement**, and **horizontal (x/z-axis) movement** are associated with said frequency data.

Weather Web Application

Flask web-application displays weather information for any city/location in the world with a google-maps style weather map and hourly temperature graph.

- **API** calls to Weatherbit.io and Openweathermap.org.
- **JSON** weather data returned is stored in an **array** of **hashmaps**, where each hashmap element in the array represents weather data for a single day of the week, such as temperature, humidity, etc. for that day.
- **HTML** results page converted to Jinja template using **Bootstrap4** for styling, to which data was passed via **Flask**.
- **jQuery** used for **DOM** manipulation to dynamically calculate and update **HTML** elements on entire page between imperial and metric unit values, using a single toggle switch.
- **Chart.js JavaScript** library used to create hourly weather graph.
- **Leaflet JavaScript** applet embedded in the page whose layers were provided by Openweathermap API and weather data was populated via hashmap array data in Flask.

Sushi Cat a Top-Down 2D Maze Traversal Game

The user plays as a cat collecting sushi in a top-down 2D maze created using turtle graphics module in **Python**. An exercise in **Object Oriented Programming**.

- **Multidimensional array** to create the playable map where player, terrain, and **object** coordinates are maintained.
- 3 **arrays** of coordinates, representing passable terrain, impassable terrain, and interactable **objects** (such as sushi).
- In each array category, the coordinate object element is divided by the sprite used to represent it, whether to appear visually as a wall, tree, flat ground, lantern, etc.
- Player/cat **object** records quantity of sushi collected and current coordinate location.
- Based on **user input** (i.e. arrow or WASD keys), player/cat **object** calculates which coordinate is requested to be moved to next. Whether or not actual movement by the player occurs is determined by if the coordinate is a member of the array of passable terrain objects.

School District Management Web Application

A **Flask** web-application for use in viewing and managing Schools, Students, Classes, and Instructors for a hypothetical School District, created for Database Design course at OSU with full **CRUD** functionality.

- **HTML/CSS** frontend whose displayed data from pages and input forms are dynamically generated from an **originally designed database** complete with entity relationship diagram (**ERD**) and database **schema**.
- **SQL queries** transmitted from webapp to the database managed with **MariaDB**. Users of the application can directly alter the database to add or remove schools, teachers, etc. and retrieve such information as well.

Animation Film Webscraper with SMS Notification

- **Python** program utilizing BeautifulSoup4 and Twilio **APIs**.
- **GET** requests to local film theatres film schedules to specifically filter for animation/anime films in real-time.
- **HTML/CSS/JavaScript** in **JSON** form is converted to unicode with BeautifulSoup4 and parsed to obtain film title, description, screening location, and screening dates for each unique film.
- Using Twilio **REST** API, the information per film is sent as an **SMS** text to a recipient/phone number of your choosing.

Website

https://john-sy.com

Github

https://github.com/DarkHorse108

Skills

- ❏ Python
- ❏ C/C++
- ❏ SQL
- ❏ JavaScript/jQuery
- ❏ HTML5/CSS3
- ❏ IA-32 Assembly
- ❏ OpenCL/OpenGL
- ❏ CUDA
- ❏ OpenMP

Technologies

- ❏ MariaDB/MySQL
- ❏ Flask
- ❏ Node.js
- ❏ Three.js
- ❏ Git
- ❏ Linux (RHEL)

Education

B.S. Computer Science
Oregon State University
(2018 - 2020)

B.S. Biology
Hawaii Pacific University
(2010 - 2013)
Colorado State University
(2008 - 2010)