2015

Set Lister

Cyril Casapao
Grand Valley State University

SetLister

HNR499: Honors Senior Project

Cyril Casapao

Winter 2015

Advisor: David K. Lange

http://www.cis.gvsu.edu/~casapaoc/setlister/setlister_index.php

Background

Local music venues may not want to book a musician if that musician has few original songs or little experience performing. Street performing offers a unique opportunity for these amateur musicians. The musician can gain valuable performing skills without needing to secure a performance at a venue.

In addition, the very nature of performing on the street to complete strangers will help build confidence and prepare the musician for handling hecklers. However, the spontaneity involved in street performance can also lead to very rewarding events such as when a small family walks by and the child starts to sing along. All these factors culminate into a learning experience that can be applied beyond a performer's musical experience.

However, street performing requires a completely different performance style than that found in a typical concert. Most often, the street performer plays solo pieces tailored to his or her musical style. The street performer will often need to place more emphasis on showmanship than would be necessary in a band setting. Since the street performer often plays alone to apathetic passerby, he or she will need to engage the audience in more than just the music.

Creating a proper setlist helps showmanship and musicality naturally work together. For instance, the performer can keep an audience on their toes by starting on "Imagine" (a soft ballad in the key of C) and switching to "Can't You Hear Me Knockin'" (a rebellious rock song also in C) after finishing the chorus. The similar keys help to smooth the transition between contrasting dynamics and ultimately help to entertain audiences.

Choosing songs for such a setlist is not a trivial task, especially for the targeted amateur musicians. Such musicians may have a limited repertoire and thus a limited pool of songs to select from. This is where SetLister comes in.

Problem

There are numerous free websites that musicians can visit to find the chords to their favorite songs. However, there are no free options for musicians who want to build setlists. SetLister solves this problem by offering performers with little technical knowledge a quick and easy way to build customized setlists.

Planning

Before development began, I knew SetLister would need to be built around a database of music. I also knew that I wanted SetLister to be freely accessible to anybody wishing to use it. Taking these facts into account, PHP was the obvious language of choice.

PHP is a popular server-side scripting language. According to W3Techs, PHP is used by 82% of websites. It is also frequently paired with the relational database management system called MySQL. PHP offers many built-in functions that make it easy to interact with MySQL. In addition, both technologies are completely open-source and non-proprietary. I personally support the FOSS (free and open-source software) philosophy and always find myself a little happier using open-source technology rather than their proprietary counterparts.

I had never used PHP before I began planning this project. Before starting development, I took a free online course offered by Udemy. However, I soon realized the constructs I learned were deprecated and destined to be removed in future versions of PHP. Luckily, the concepts carried over so I simply read relevant chapters of Paul DuBois' "MySQL Cookbook" along with the official PHP documentation to familiarize myself with the most recent techniques and constructs.

Development

The beginning stages of development were admittedly rather tedious. Before any coding began, I had to find information on 100 songs to be entered into the database. In addition to the chords used in the algorithm, I decided to include extra information such as release date and artist name. I tried to pick well-known songs that diverse audiences would likely recognize. I also included songs that have proven popular in my own experience as a street performer.

During this stage, I also developed sketches of the user interface on paper. It allowed me to see what I would need to include in the end product in addition to letting me plan how different pages around the site would be linked. I used Dreamweaver to create a base template that would be used by all pages on the website to create a unified and professional feel to the application.

Once the data entry was complete, I began writing PHP scripts that would provide a user-friendly interface to the database. Naturally, this was the most demanding aspect of the project so I allotted most of the development cycle to this process. The scripts that provided the interaction took less time than initially estimated.

To balance this out, the setlist creation algorithm took longer than expected. In my opinion, PHP's syntax for object-oriented programming is more cluttered and quirky than that of other object-oriented languages such as Java or Ruby. This initially slowed down development because of simple mistakes I made such as forgetting to include the ***this*** keyword when referencing methods within a class.

I met weekly with Professor Lange to discuss progress and improvements. This was a tremendous help and broadened my view of the project. I found that I would spend so much time developing the algorithm that I left the user interface slightly neglected at times. These meetings with Professor Lange helped keep me focused and on track. I also learned to keep usability in mind during the development process.

How Does It Work?

In order to allow me to focus on the technical aspects of development, I decided to use very basic music theory to build my setlist creation algorithm. Each song in the database has one or two chords associated with it, usually representing the first chord in the song, the first chord of the chorus, or other important chords such as those marking a transition. In other words, the selected chords are points in the song that would make good transition points into or out of the song.

A Generic View of the Algorithm

Any song $\boldsymbol{M}$ in the database has two chords ($\boldsymbol{m_1}$ and $\boldsymbol{m_2}$) associated with it. Only $\boldsymbol{m_2}$ may be null, which means $\boldsymbol{M}$ only has one associated chord. If $\boldsymbol{m_2}$ is null, we simply ignore it during execution of the algorithm. $\boldsymbol{M}$ is considered "linked" to another song $\boldsymbol{N}$ (with associated chords $\boldsymbol{n_1}$ and $\boldsymbol{n_2}$) under the following conditions:

1) Any chord $\boldsymbol{m}$ is the major version of any chord $\boldsymbol{n}$
2) Any chord $\boldsymbol{m}$ is the minor version of any chord $\boldsymbol{n}$
3) The root of any chord $\boldsymbol{n}$ is a perfect fifth above the root of any chord $\boldsymbol{m}$

In this way, we create a graph of songs where each song is a node and with connectivity decided by the conditions stated above. Thus, we can use a recursive depth-first search starting at any given node to traverse the graph and accumulate a list of linked songs. Each song is guaranteed to be included only once since we mark songs as visited once we add them to the setlist. This is important because we do not want to play the same song twice during a performance.

Future Development

In its current state, SetLister's main functionality works as it should. I originally wanted to include the ability for users to make lists containing only specific genres but with such a small data set, it would not create sizeable lists. Some tweaking could be done to allow for this functionality if the data set were to expand.

On that note, if users were allowed to suggest songs using a specialized input form, the data set could be expanded. However, this would require much dedication outside of school since I would need to look up information about the requested songs and decide whether or not the songs were appropriate for SetLister.

The algorithm could also be expanded to include more chords. SetLister only deals with chords with natural roots. In other words, chords like C# or Gb are not considered in the algorithm.

Reflection

Developing SetLister was a valuable experience to me. PHP coding is a very marketable skill in industry and Grand Valley does not offer students many opportunities to study it. Beyond coding ability, I also learned the importance of keeping the user in mind during development. I believe this way of thinking will allow future projects I work on to be more user-friendly. I consider myself very fortunate to get hands-on experience with the SetLister project. This experience has already proved to be beneficial: I will be interning at a local company as a PHP developer for the summer semester.