Music You Know

A website that plays songs that you forgot you knew every word to

By: Jacob Feldman

Andrew Mumm

Table of Contents

Overview

Music You Know is a small website that plays music that you used to hear all the time but have since forgotten about. It accomplishes this by playing songs from the time when you were from ages 12-18 (the time in most people's lives where they develop a "taste" in music). We've accomplished this using Python scripts, the Django framework (to build a server), and webcrawling techniques to pick a random song off of the AT Top 40 from a random year (while you were an age from 12-18). It then embeds a YouTube video of that song in our webpage; we used top40charts.net to obtain the records of Top 40 songs and YouTube embed links for them.

New and Complex

To create our website, we created a web server using the Django framework for Python. The Django framework requires a variety of different files that work together to make a server; while this organization helps simplify construction of large websites, it effectively complicates the construction of simple websites. We also used Python to write some scripts that web-crawled through top40charts.net and YouTube. Prior to this project, neither of us had any experience with web-crawling or Python, both of which are complex. We used the urllib and HTMLParser Python libraries to open a connection to a webpage and parse that webpage for the html elements we needed. We had to make sure our scripts only selected the correct element out of all the similar elements in a usually very large html document, and the script has to do it at each step of the multistep web-crawl.

Bloom's Taxonomy

Analyzing:

Our project consists of several distinct pieces that come together to form a website. We wrote a script in python that acts as a web-crawler. This web crawler searches 2 websites for information. The web crawler is initially supplied with a year to search for. The web-crawler is trying to look a for a top 40 song for that year. When it finds that song it will then search for a video corresponding to that year. After that it returns the YouTube embed code corresponding to that video.

We wrote an HTML and Javascript file that serves as the frontend of our website. On the website, the user enters in the year that they were born. The Javascript sends this year to the server. The server adds a random number between 12 and 20 to the year. The server then sends the year to the python web-crawler so it can look for the embed code for the given year. When the HTML page receives the response back from the web-crawler it displays the video with the embed code.

We tied the pieces together by using a python framework called Django that helps to implement a server. Django runs the HTML page on the server and allows for communication between the webpage and various functions and scripts. Django is the glue that holds the whole website together. It ties together our frontend HTML and Javascript with the backend of the web-crawler.

In summary, Django runs a server for the webpage, and the user interacts with the webpage that was written in HTML and Javascript. Based on the users input Django tells the python web-crawling script to execute and send the results back to the HTML page. The webpage then uses this information to insert a video into the page for the user to watch.

Evaluating:

By the end of the project the user should be able to navigate to the web-page, enter their birth year, and watch a video for a song that they would likely know based on when they were born. We were originally going to try to use AJAX to have our python and HTML files interact with each other, but we were not successful in figuring this out. This is when we decided that it would be better to use a server. We learned about Django and decided that this would be best for our needs. Our website design works as intended and meets the original specifications that we intended.

Creating:

The first part that we created was the python web crawler. The web crawler is broken up into four stages that systematically returns an embed code for a YouTube video to be used on our website. On the first stage of the web crawl our script searches a web site at http://www.top40charts.net/ . This website has a list of the top 40 songs of each year since 1950. We give our script a year and the script navigates to the top 40 list for the year that we provided. On this page, there is a list of 40 songs and who wrote them along with a link to YouTube for that song. The second stage of our script randomly selects one of the top 40 songs for the provided year and navigates to YouTube. On the YouTube page there is a list of videos for the song that is selected. On the third stage, we have our web-crawler select the top video link in the search and navigate to that videos page. This page is just the normal page you would see when watching a YouTube video. The fourth stage of the web-crawl searches the page for the embed code of the video. Once the embed code is found the python script returns the embed code as a string to the location the script was run from.

The second part of the website that we developed was an HTML, CSS, and Javascript page. This page is designed to ask you for the year you were born with a text box provided for your input. There is also a button that you can click on in order to submit your answer. The Javascript script then takes that year and uses it to redirect the webpage to another with the same URL as before except that it now ends in the year that you entered. At this point we needed a way to tie together our webpage with the web-crawler that we designed.

For the third part of the project we needed a way to run our webpage, tie together our python script with our webpage, and execute the code on a button click. To accomplish this task, we decided to use a python framework called Django. Django is a framework that helps you build a server and execute code based on user input. Django recognizes URL patterns that you give it and uses those patterns to execute functions and decide what to display. When we have our users enter in their birth year, we change the URL and have it end in the year entered. Django recognizes the change in the URL pattern and executes a function that we tell it to. This function take the birth year provided and adds a random number between 12 and 20 to it. This is so that the user will get a song from a year that they are familiar with. The function then calls the python script and passes as an argument to it the new year that we calculated. The python script will then execute the web crawl and return the embed code to be saved as a string. This string is then sent to the HTML page and used to insert the selected video into our webpage for the user to watch and listen to. The user can then press the button again to be provided with a new video.