Description
This project is a visualization of last.fm stored user data. It creates an avatar of a user based on their musical selection from data scraped form the Last.fm website. Then the program selects 10 friends/neighbors of that user and creates avatars for them. All of these avatars are shown in one window. The user is also able to show pie chart representations of user instead of avatars. The pie charts show what each musical genre accounts for in a user’s favorite artists. Finally, all of the avatars can be made to randomly walk about the screen together on a grassy hill.

Audience
The audience for this project is college age music aficionados. People who use social websites such as Facebook or Myspace and often listen to music on their laptops or computers will benefit from this visualization. Graphing neighbors of a user will lend the application to social networking and broadening users’ horizons. We image people between the ages of 16 and 25 would be willing to use this visualization, but it is usable by anyone with a Last.fm account.

Purpose
There are two purposes why someone may wish to use this visualization. The first is for meeting new people with similar musical tastes. Our program uses a feature of Last.fm called “neighbors”. Last.fm finds people with similar musical tastes as a user and lists some of them in a group called neighbors. Our visualization extracts this list of neighbors and visualizes them along with the main user.

The second reason to use this application is for self introspection. An avatar of the user is created which is meant to reflect that user’s musical preferences. This allows a user to reflect upon the music they listen to and see whether it is how they expected. Perhaps, for example, they will find their musical tastes are softer than expected.

Work Progress
We started with a simple brainstorming session which yielded the following design idea. The design is simple, yet functional. The bottom of the screen would list a user’s friends and neighbors. The middle section would have larger profile views of the main user and one selected friend. Finally, the top would house buttons for logging in and toggle switches. We planned out the amount of time needed to draw the avatars, create the button objects, and design the Last.fm webcalls. The first concept for the avatar follows.
We started working the Friday before the project was due. Nick drew all the picture for the avatars and then wrote the code to load and animate them. Nigel wrote the code for performing Last.fm API calls and parsing the responses as well as coding button functionality.

**Implementation**

**Last.fm API**
Our project was written in C# using Microsoft’s XNA Game Studio. We use Last.fm API calls to generate our data. Last.fm provides information on users’ musical preferences and friends as well as information on musical artists. We make web requests and then parse the XML results. To determine what a
person’s musical tastes are, each artist they listen to is converted into a specific genre. Last.fm gives a list of a user’s most popular artists and it provides a list of user created tags for each of those bands. Our program parses the tags for an artist and determines a genre, and then this information is cached in a file. After a list of genres is collected for a user, a pie chart is created representing each genres percentage in the collection.

**XNA Game Studio**

In order to do the actual visualizing of the data, we used Microsoft’s XNA Game Studio. While XNA Game studio is a development framework that is designed primarily for the creation and production of games for both the PC and for the Xbox 360, its ability to easily render, translate, and rotate 2D sprites made it ideal for our project. In addition to being able to render our custom 2D sprites with ease, the fact that my partner and I were both already very familiar with Microsoft’s XNA game studio made it the most logical choice of a development framework.

Because an avatar would, based on the musical preferences of the person who was represented by that avatar, be wearing bits and pieces of various genre-defined outfits, we needed a good (and – hopefully – simple) way of easily changing an avatar’s clothing. In order to do this, we eventually decided that a method of drawing each base outfit (e.g. the hard rock outfit, the hip hop outfit, etc.) as five separate images – one for each of the three interchangeable body parts (i.e. one for the head, one for the torso, and one for the legs) as well as one for each of the avatar’s two arms. Choosing to draw the head, torso, and legs separately made the process of creating an avatar for a person whose top three music genres were pop, soft rock, and dance as simple as swapping out the head, torso, and legs of the base (i.e. naked) avatar with those of the pop, soft rock, and dance avatars respectively. Our decision to draw the arms separately from the torso was born out of the fact that we wanted to be able to easily rotate an avatar’s arms about his shoulder joints. Drawing the arms as separate, individually rotatable objects was, by our determination, the simplest way to allow the arms to rotate while also keeping the quantity of needs-to-be-created art down to a minimum.

![Figure 3: The Final Versions of the Avatars (From Left to Right: Hard Rock, Soft Rock, Country, Dance, Hip Hop, Pop)](image-url)
Drawing the pie charts that showed how much of each musical genre the corresponding person listened to also presented a moderate challenge. While we had many times before drawn static, unchanging images in XNA game studio, my partner and I had never needed to actually construct and image using the tools available in XNA game studio (externally drawing all possible pie chart configurations was not practical). We ultimately decided to draw the pie charts using a primitive renderer to draw a triangle fan, which is a means of drawing a string of connected triangles that all share a single, central vertex. Each triangle was drawn as a isosceles triangle where the angle at the triangle’s peak (i.e. that angle which is not equal to the other two) was one degree. In this way, we were able to construct objects that appeared to be circles out of 360 of these very tall, thin triangles. Also, each of the 360 triangles was colored in such a way that the resultant pie chart correctly displayed the colors representing the percentages of listened-to musical genres.

![Musical Tastes](image)

<table>
<thead>
<tr>
<th>Artists</th>
<th>Genre</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beatles</td>
<td>Soft Rock</td>
</tr>
<tr>
<td>Pearl Jam</td>
<td>Hard Rock</td>
</tr>
<tr>
<td>Kanye West</td>
<td>Hip Hop</td>
</tr>
</tbody>
</table>

*Figure 4: Pie Chart and Corresponding Table of Artist-Genre Assignments*

**Related Work**

**Mii Channel**

In our application, we wanted a mode where the avatars could move around in a space and ‘socialize’, thereby giving more life to the program than could be given simply by displaying a series of avatars and pie charts. For inspiration on how to implement this aspect of our program, we looked to the Nintendo Wii’s Mii Channel. In the Mii Channel, all of the Miis on one’s Nintendo Mii wander about in a small space and engage in ‘conversation’ – represented by small speech bubbles – with their fellow Miis (and also occasionally fall asleep). The final implementation of our own ‘avatar socialize’ mode very much resembled the interactions of the Miis in the Mii Channel. While no speech bubbles appear above the heads of our avatars (nor do they ever grow tired and feel the need to doze), our avatars do wander randomly in a space and sometimes, when they happen to be close to one another, appear to be socializing. Thus, while somewhat simple and limited, we believe that our ‘avatar social space’ was a success.
Future Work

While we believe that our application is, in its current form, a worthwhile and entertaining application, we also can see that there is much room for expansion and improvement. The primary aspect of our application that needs to be improved before any other advances are made is the login/search feature. In its current state, retrieving the artist information of a given user on Last.fm (providing that that information has not been locally cached) takes upwards of 10 minutes. While we were unable to pinpoint the exact reason for this large amount of necessary time to collect a user’s relevant data, we believe that much of the problem lies in the fact that the Last.fm servers allow only a small number of data queries per second and also require that all such queries be cached locally for an extended period before an identical query is made. Thus, in order to improve the login time, we most likely should develop and different (and better) strategy for querying the Last.fm servers and caching the data. The second area that needs improvement is the way we have chosen to classify the music to which one listens. We currently categorize every musical artist in such a way that he or she falls into one of only five genres (pop, hard rock, soft rock, dance, and hip hop). This pool of genres is obviously quite limited and should be expanded in order to improve the overall accuracy of the visualization. Along these same lines, it would also benefit the application greatly if we could increase the number of genres displayed on a given avatar (i.e. more that only three genres – one for the head, one for the torso and arms, and one for the legs). Both of these improvements (increasing the size of the genre pool and also increasing the number of genres displayed on the avatar) would be relatively simple and, aside from the rather significant amount of time it would take to create the art for each musical genre, easy to implement. Also, the application could be improved by increasing the number of types of data that can be displayed. Currently, the avatar and pie chart are based only upon the top artists to whom a given user listens. If this could be expanded so that a user could toggle between a means of displaying data based upon the top artists or, additionally, the top songs to which a person listens, we believe that the application would be greatly improved.

What We Learned

We talked with a few Last.fm users and showed them our visualization. They reacted with a desire to corrupt their records in order to change the visualization of their data. One person wanted to leave David Bowie playing all night on mute. This is an expected, but unwanted reaction.

It is difficult to find a small set of musical genres which can represent all music. This is in fact the Set Cover problem and is NP-Complete for a computer to calculate. A second difficulty is finding a way for the computer to learn about what type of music an artist plays. Our program relied on mapping keywords from user generated artist Tags to one of six musical genres. This solution worked well enough, but it relies on Tags created by users. This runs into the issue of how exactly an artist fits into a specific genre. Many artists have several characteristic sounds which fall into multiple genres of music. For example, Fleetwood Mac began their career as a Blues/Folk band. Later they transitioned into a more classic rock/pop sound.
Screenshots

Figure 5: Screenshot of the Data-View Screen in Pie Chart Mode

Figure 6: Screenshot of the Data-View Screen in Avatar Mode

Figure 7: Screenshot of the Socializing Avatars Mode