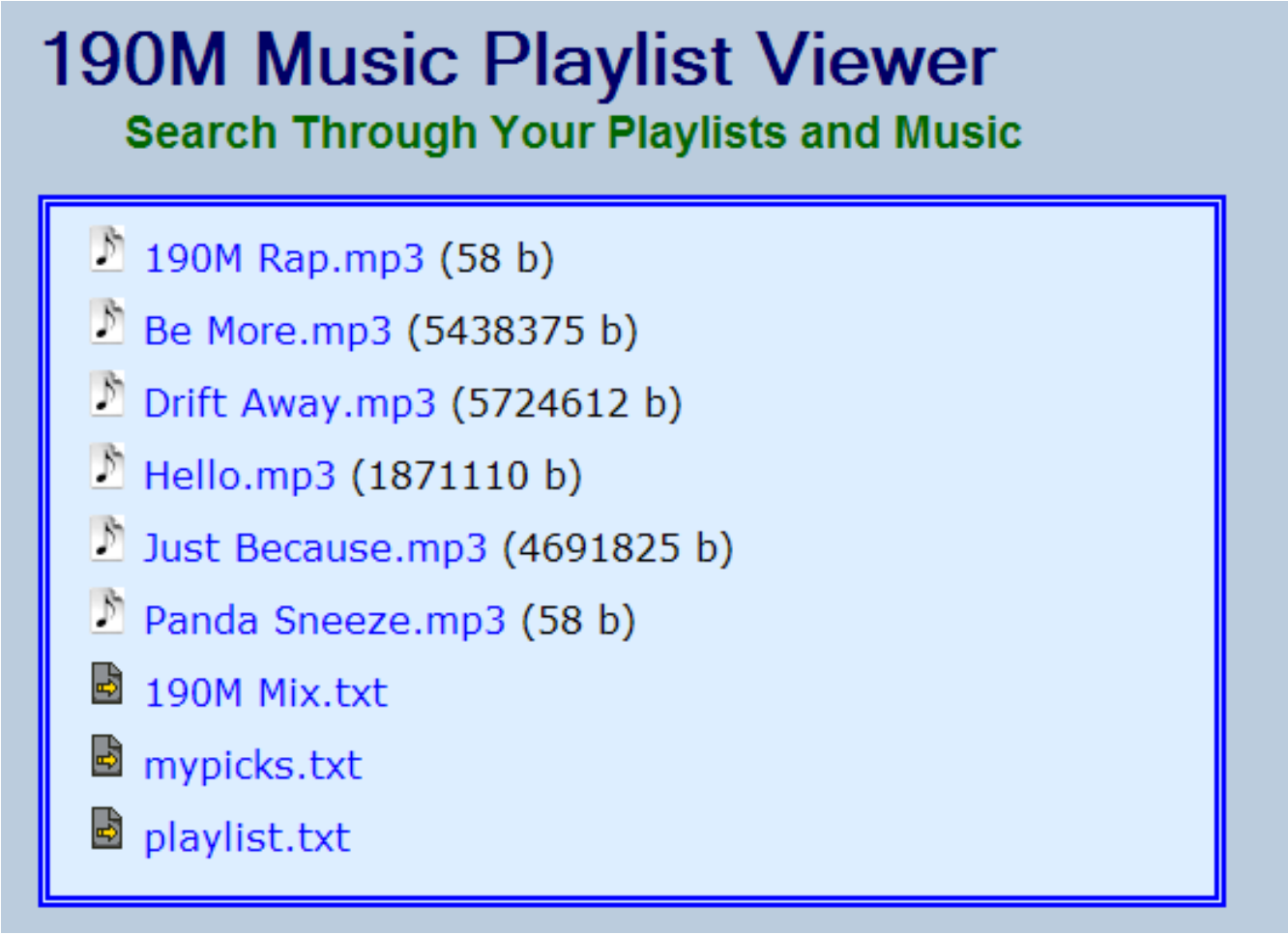
Internet Programming Lab 3: PHP Basics

MP3 Music Library

Brian has written a page named `music.html` (right-click and select "Save Link Target As...") that contains a display of all the MP3s he has on his computer. He has already written the HTML code and the styles for the page, `viewer.css`. (You don't need to do any CSS for this lab.)

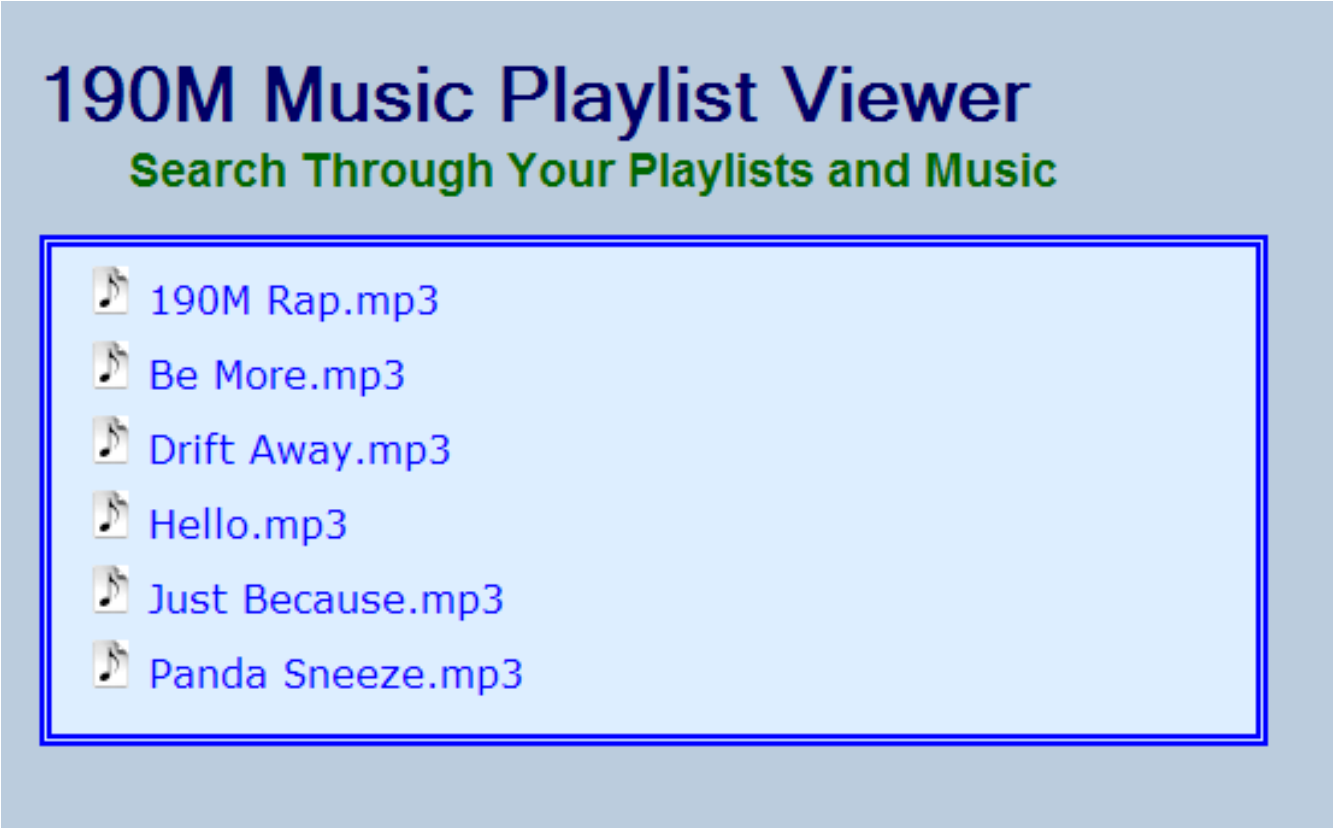


However, the page is not very flexible, because he must manually edit it every time he adds songs to his collection. In this lab, you are to turn this page into a dynamic PHP page in a file named `music.php` that will list the MP3s and will always display the songs currently on the computer.

Exercise 1: Display MP3 Files (roughly 20 minutes)

The first task is to convert `music.html` into a PHP page that displays Brian's MP3s. Examine the files in the `songs/` folder and display each `.mp3` file in the format currently shown in the `music.html` page. Each one becomes a list item (`li`) with a `class` of `mp3item`. Each song's title should be a link to download that song file from the computer.

Your page should look like this:



It may help you to remember that the `glob` and `scandir` functions list files in a directory, and that the `basename` function extracts the file name from a larger path string.

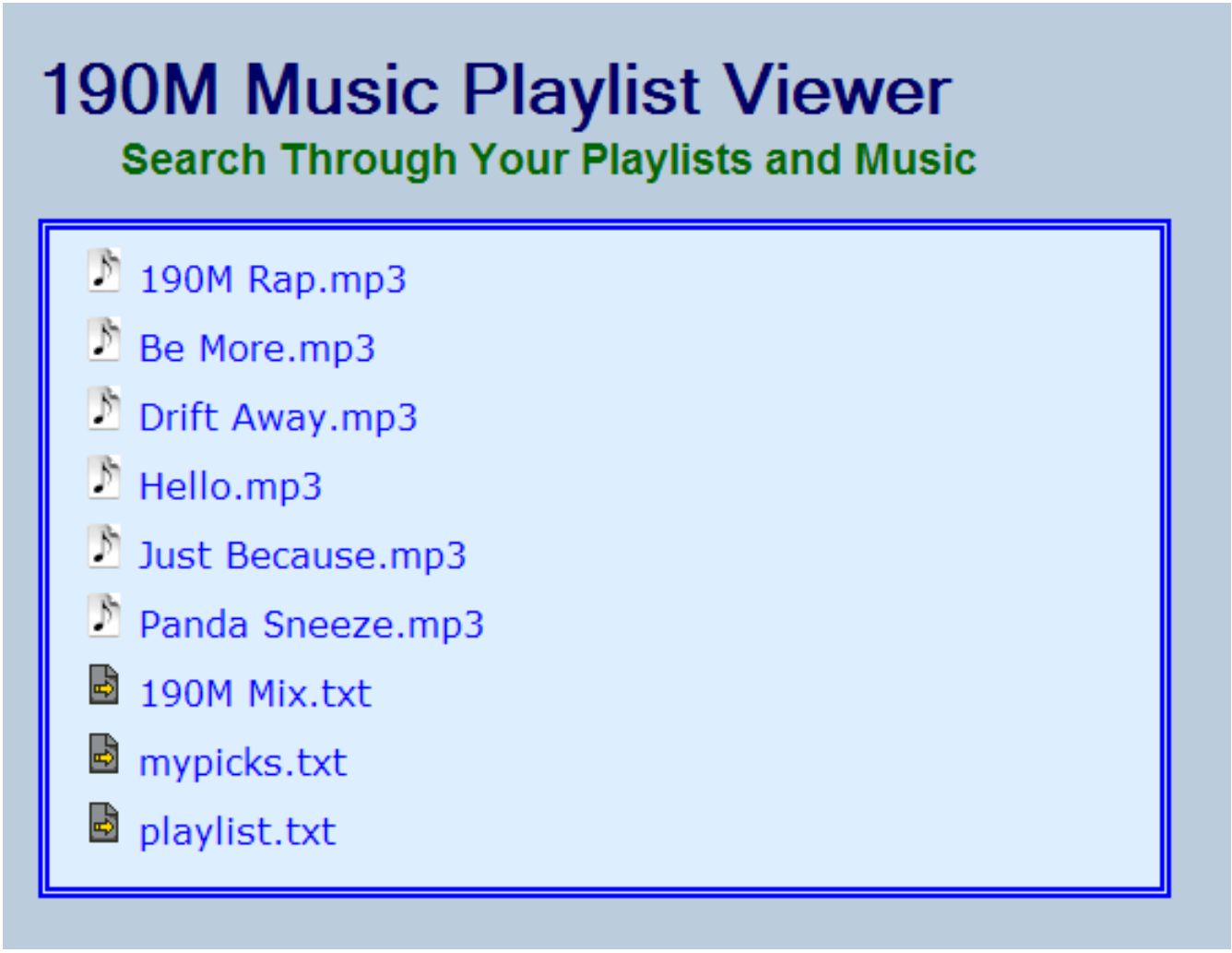
You may want to download this list of MP3s and playlist files for testing:

- songs.zip

Exercise 2: Display Playlists (roughly 10 minutes)

Brian's music collection also includes *playlists*, which are text files containing lists of his favorite songs. For this exercise, modify your PHP code to display any playlist files, which are `.txt` files in the `songs/` directory. They should display very similarly to the `.mp3` files, except they use a `class` of `playlistitem` so that they have a different icon. For now, a playlist's link is the same as an MP3's link: It downloads the file.

Your page should look like this:



Exercise 3: View Songs in a Playlist (roughly 15 minutes)

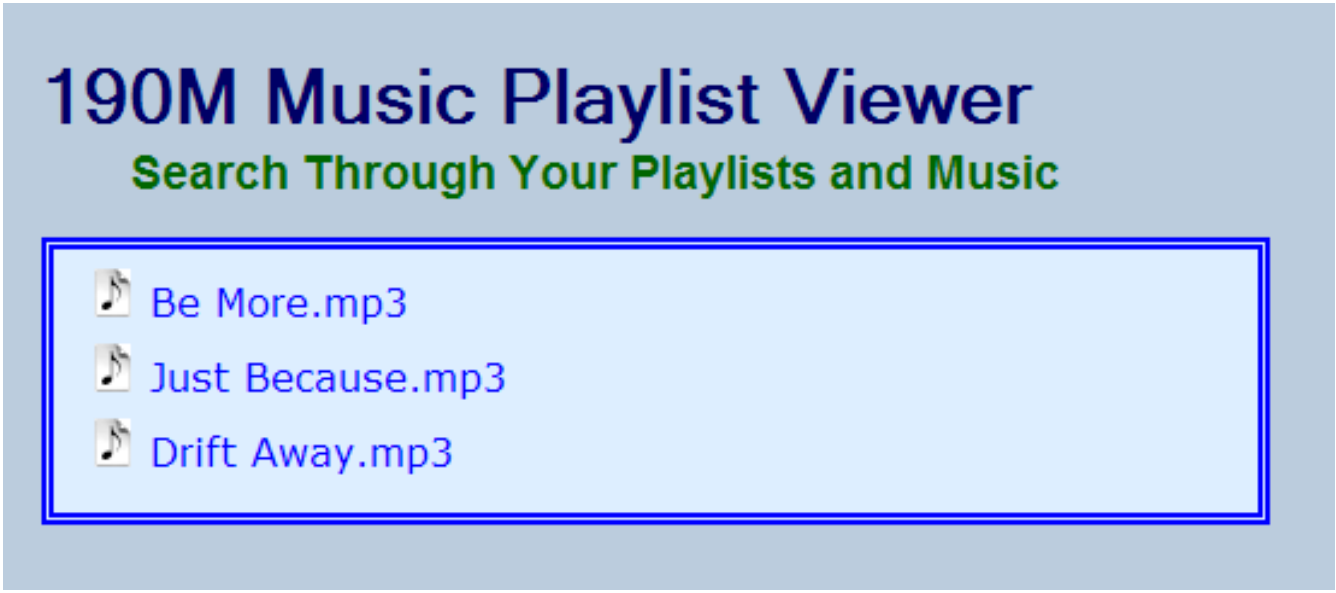
Next you should modify the behavior of your page so that it can optionally display just the songs that are listed in a certain playlist. A playlist is just a text file, where each line of the file contains the file name of an MP3 that is part of the playlist. For example, the file `songs/mypicks.txt` has the following contents:

```
Be More.mp3
Just Because.mp3
Drift Away.mp3
```

For this exercise, change your PHP code so that if your page is given a *query parameter* of `playlist`, it will read the playlist text file with the given name and display its songs only. For example, if your page is displayed with the following URL:

```
music.php?playlist=mypicks.txt
```

Then your page should look like this:



You may assume that any songs contained in a playlist actually do exist on the computer.

Recall that query parameters are put into a global PHP array named `$_REQUEST`. To access the above query parameter, you should write a line of code such as the following:

```
$playlist = $_REQUEST["playlist"];
```

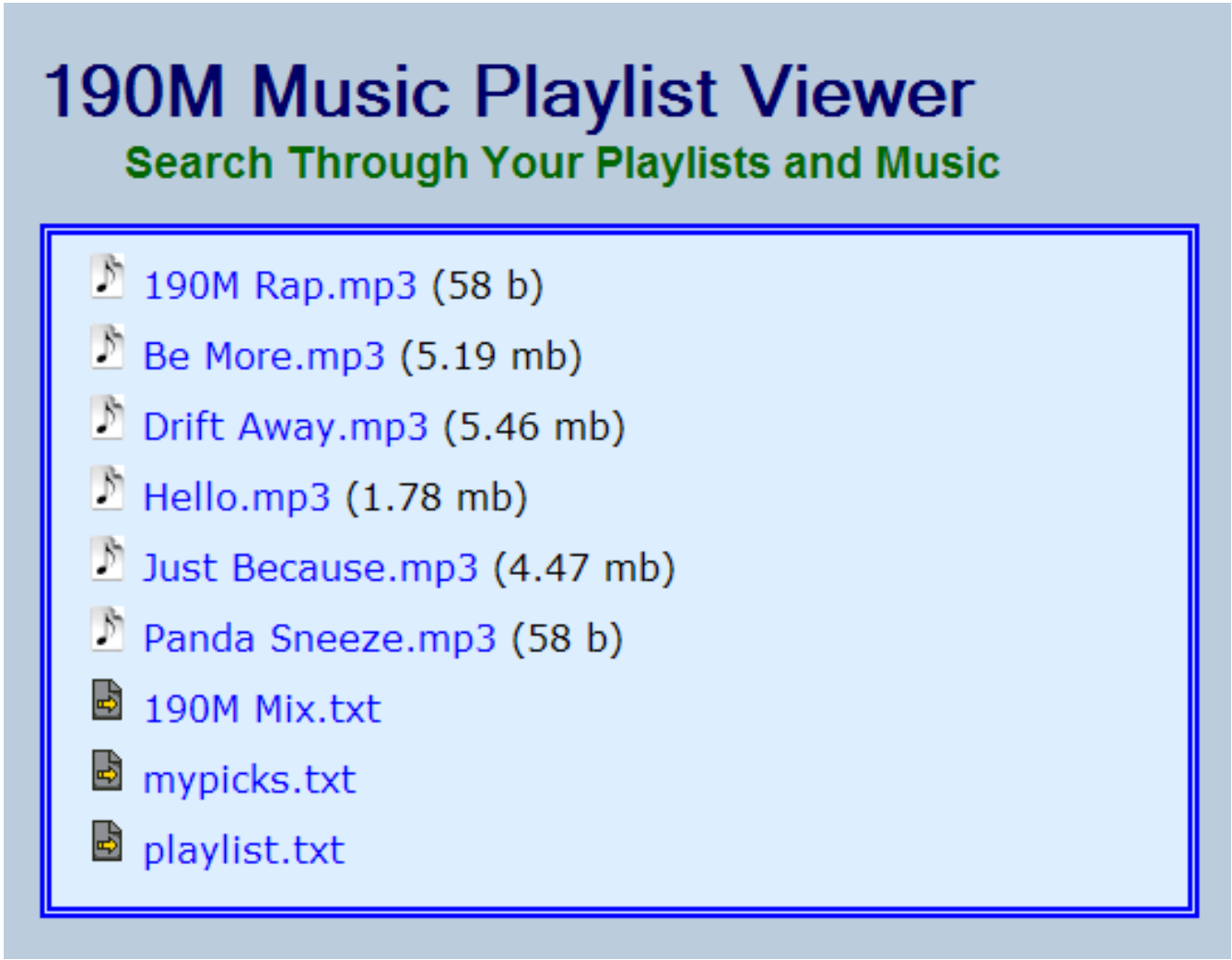
If no query parameter is given, your page should use its previous behavior, displaying all `.mp3` and `.txt` files. You can test whether a particular parameter has been passed by calling PHP's `isset` function.

Exercise 4: Song Size Info (roughly 10 minutes)

If you have time, add some information to the end of each song shown on the page. Display the song's size in the appropriate unit. If the song is 0 to 1023 bytes in size, display it as bytes. If it is 1024 to 1048575 bytes in size, display it as kilobytes (multiples of 1024 bytes). If it is 1048576 or more bytes, display it as megabytes (multiples of 1048576 bytes).

If displaying kb or mb, you should round your value to the nearest hundredth. You may want to review the documentation for the `round` function.

Your page should look like this:



Exercise 5 (for 1337 h4x0rz only): Extra Features

If you finish all of the above exercises, you can try adding any of the following optional extra features to your page:

- **Back link:** When the user is viewing a playlist, it would be nice to have a link that goes back to the main overall page to display all songs again. Add such a link to the top of your page.
- **Support real `.m3u` playlists:** The actual playlist format used by MP3 player software, `.m3u`, is actually just a text file like our text playlists were. The only difference is that in a `.m3u` file it is legal to have comment lines that begin with a `#` character. For example, here are the contents of a file `mypicks.m3u` that is equivalent to `mypicks.txt` shown previously:

```
#EXTM3U
#EXTINF:177,Paul Dateh & Oren Yoel - Be More
Be More.mp3
#EXTINF:291,Christopher Toy - Just Because
Just Because.mp3
#EXTINF:238,Magnetic North - Drift Away
Drift Away.mp3
```

  Modify your page so that it lists playlists as `.m3u` files instead of `.txt` files. When your page is reading such a file, you'll need to skip over any comment lines to show just the songs in the list. You may want to review the string functions from the book or slides to see how to search a string for a given substring.

- **Shuffle:** Add another optional query parameter to your page named `shuffle`. When `shuffle` is set to `on`, you should display the songs in a random ordering rather than in their default sorted order. For example, if the page were requested in the following way:

```
music.php?shuffle=on
```

  Then the songs would display in a random order. You may want to review the array functions from the book or slides to see how to rearrange an array's elements.

- **Sort by Size:** Add another optional query parameter to your page named `bysize`. When `bysize` is set to `on`, you should display the songs sorted from largest to smallest size rather than in their default sorted order. For example, if the page were requested in the following way:

```
music.php?bysize=on
```

  Then your page should display the largest songs (such as `Drift Away.mp3`) first and smallest songs (such as `190M Rap.mp3`) last.

  The appearance is unspecified when both `shuffle` and `bysize` are set to `on`.