Assignment: Spotify-Like-App Part 1

Start Assignment

Due Nov 3 by 11:59pm **Points** 200 **Submitting** a text entry box or a file upload **File Types** pdf

What to do

Write a program app that behaves like Spotify and plays music.

The requirements for this part of the assignment are to:

- Write a text based menu with the following options for the user
  1. [H]ome
  2. [S]earch by title
  3. [L]ibrary
  4. [Q]uit
- Select and download a set of 10-15 or more audio files of your choice from https://freemusicarchive.org/ (https://freemusicarchive.org/)

  Although you will only download and use audio files in this assignment, your software should be designed to easily scale (for example by changing a constant) to handle a potentially infinite number of audio files.
- For each audio file, your app should save the following fields for each audio file and display this information when the audio plays.
  1. title
  2. artist
  3. year
  4. genre
  5. filePath (the path to the .mp3 file in your git repository on your computer)
- Your [H]ome menu can organize and display your audio tracks however you wish. It can also play audio files if you wish. But, this is not required
- Your app must allow a way to play audio files in the following menus
  1. [S]earch by title
  2. [L]ibrary
- All of your sub-menus can & should be simple text-based menus. For example, selecting a number 1-15 to play an audio files is perfectly acceptable.

Task 1

- Create a new GitHub repository by using GitHub desktop
- Perform a GitHub commit and publish to publish your repository on GitHub.com

Task 2

- Create a new Maven project within the GitHub repository.
- Test and make sure your App.java (hello world) runs successfully.
- Perform a GitHub commit with a short concise commit message.
- Perform a GitHub push to save your work to GitHub.com

Task 3

- If you wish to add json support into your app, you will need to do the following.
- Go to **Google's json maven repository (https://mvnrepository.com/artifact/com.googlecode.json-simple/json-simple/1.1.1)** .
- Edit your pom.xml file to add Google's json library to your project.
- Perform a GitHub commit with a short concise commit message.
- Perform a GitHub push to save your work to GitHub.com

```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.11</version>
    <scope>test</scope>
  </dependency>
  <!-- https://mvnrepository.com/artifact/com.googlecode.json-simple/json-simple -->
  <dependency>
    <groupId>com.googlecode.json-simple</groupId>
    <artifactId>json-simple</artifactId>
    <version>1.1.1</version>
  </dependency>
</dependencies>
```

Task 4

- Download the **example code and data files** (https://sjeccd.instructure.com/courses/33703/files/5734933/download?download_frd=1) .
- Place the example code and data files into the repository. The example code java should be placed in the same directory as your Maven's default App.java file.
- Perform a GitHub commit with a short concise commit message.
- Perform a GitHub push to save your work to GitHub.com

Task 5

- Edit, test, and run the example code (given below) using Visual Studio Code and get it to work on your computer.
- This often involves changing paths and figuring out technical issues on your computer.
- Perform a GitHub commit with a short concise commit message.
- Perform a GitHub push to save your work to GitHub.com

Task 6

- Continue to solve the assignment and work on the features.
- The data should be read from the json file (see my example) and stored into a Dictionary (hash map) such that given a student's name as input the program prints the students birthday.
- As you work and make small changes to your code, perform a GitHub commit and include a short concise commit message to continue to help build up your Git history through development.
- Perform a GitHub push to save your work to GitHub.com
- Repeat this task until you complete the homework assignment.

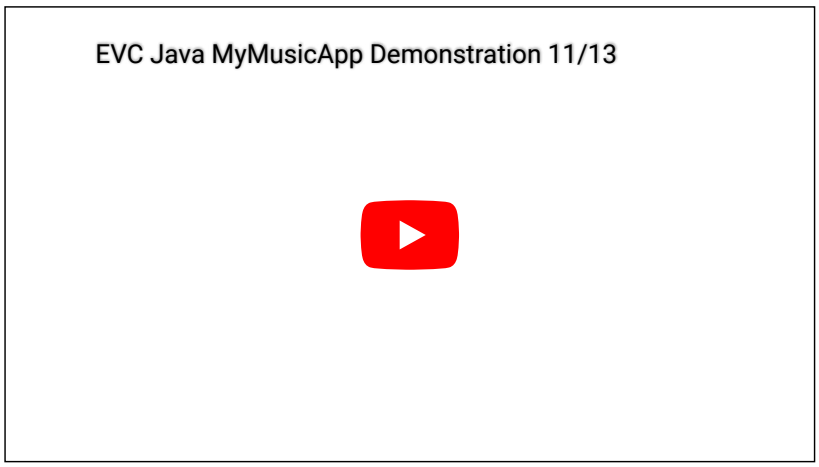
Tips & Hints

- Does the example code not work on your computer?
  - If not, you probably have not changed the path in the code so that the example code works on your computer, or you have some other file/folder organization problem on your laptop.
  - If you do not understand how files are stored on your computer and how paths work, please do consider doing the extra credit Module on "Folders and Directories". I made this module to help students who need this knowledge.
  - There are also 100's of YouTube videos on this topic that you can view.
- There are many libraries to play audio files with Java. You can use whatever audio library you wish.
- Java Sound Libraries
  - I am using the standard javax.sound libraries. It only supports .wav files that are large files (not .mp3). So, if you use my technique, you will need to use a .gitignore to not bloat your repository.
  - To use this technique, download a free tool that convert .mp3 to .wav or find some example wav files that you can use. If you are on a windows pc, you will need to convert the audio files to 8 or 16 bit audio files. There are many free app to convert audio file formats on the internet.
  - I have provided 10+ audio files that you can use if you dont wish to select and convert your on. Those files are in the Canvas Files->Assignments->SpotifyLikeApp->example audio folder.
- An excellent start would be to create a “play” function that works for one of my cropped audio file. Then, to build out your menu system with this 1 song. Adding the other 9-15 audio files should be straight forward from there.

- Make friends in class! 😊 Reach out to other students and work together. Please do collaborate with your classmates. Help each other and look at each other's code. This assignment is not easy/trivial. Making friends and collaborating is a great thing for developers & students to do. I will not penalize you for this.
- Here are some links that may help
  - How to play an Audio file using Java (https://www.geeksforgeeks.org/play-audio-file-using-java/#:~:text=Following%20steps%20are%20to%20be%20followed%20to%20play%20a%20clip%20object.&text=getAudioInputStream(File%20file),.()%20method%20of%20Clip%20interface)
  - Java Sound Technology (https://docs.oracle.com/javase/8/docs/technotes/guides/sound/index.html)
  - Package javax.sound.sampled (https://docs.oracle.com/javase/8/docs/api/javax/sound/sampled/package-summary.html)

Example Student Demo Video

This is a video from Sneha Revanur (student) is an excellent example of a demo video that she screen recorded. Observe how she walks through each feature and explains it in detail.



How to submit your assignment

Step 1 - Use the example assignment template

- Create your own an empty Google Doc using your google account.
- Students must use this **example assignment template (https://docs.google.com/document/d/1cdgoY3kthBdQxMGDZgK5FIgWOQISGI0PY8TZ5NjEDoA/edit?usp=sharing)** to submit this homework assignment. To do this, simply copy/paste the information from the example assignment template Google Doc into your own Google Doc file.

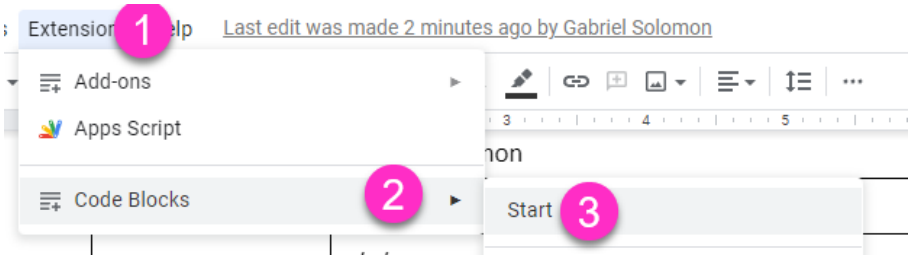
Step 2 - Add your assignment information

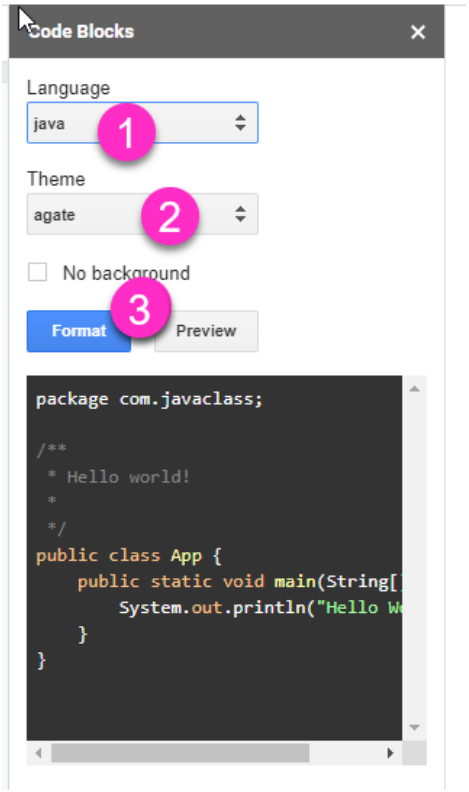
- Replace my information in the template with your work from this assignment.
- **Submit ONE SINGLE file as a pdf that contains the following:**

Step 2.1 - Copy/paste your code into the Google Doc

- Copy/paste your code from Visual Studio Code in to your Google Doc assignment submission file.
- Format your code using the Code Blocks Extension. Go to Extensions->Code Blocks->Start. Then, in the code blocks window area, Select Preview, Select the language, Select the "agate" format, then select Format. See screenshots below.

- Your code should appear as it does in VS Code. All formatting and information and spacing should be "proper" and correct as it looks in Visual Studio Code.
- No manipulation nor by-hand formatting of code should be necessary.





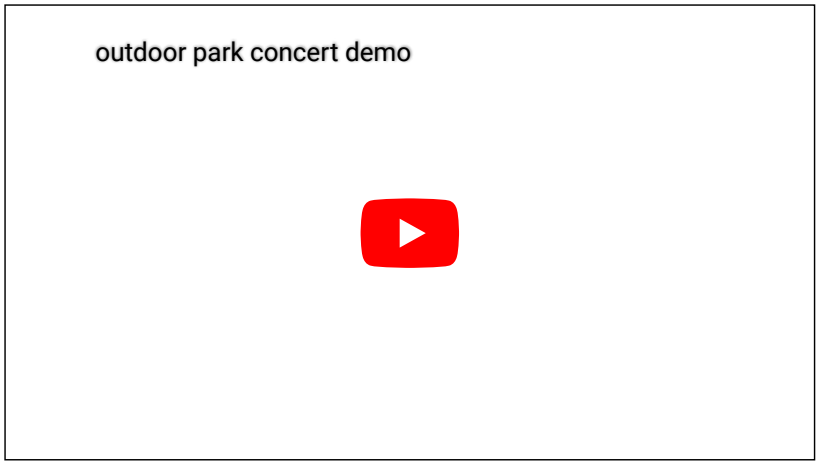
Step 2.2 Create a Demo video using these Guidelines

Below is some guidance and an example demo video.

1. Students should not show their code in their YouTube demo video.
2. Students only need to show the software running in their YouTube demo video
3. Demo video should demonstrate any features implemented.
4. The video should be short and concise.
5. Students need to talk over their demo (using audio) to explain what they implemented and how their software works and what it does.
6. If students do not wish to talk, they can use text, arrows, titles, and other annotations on the video to explain their work.
7. Students do not need to appear in their video. I do not need to see your face. I just need your explanation.
8. Students should not worry about making a "nice or fancy" presentation. Students should simply and very briefly explain the software features. I am not asking for a presentation.
9. Your video should be high enough resolution that a person can read and see it what it does.
10. Always watch your video **after** you upload it to YouTube, to make sure it appears as you wish.

Example: Make your demo video similar to this example demo video

Notice how in this video I simply run through the features of the software and demonstrate the features. This is all students needs to be done. Students do not need to do more. Please keep your demo videos simple.



Step 2.3 - Create an unpublished YouTube video and add the link into the Google Doc

- Create an **unpublished** YouTube demo video of your software working as follows:
  1. Use the demo video guidelines as described below.
  2. The demo video should show your software running and working. You will need to "demo" (demonstrate) your software and all the features and functionality you built.
  3. You can also talk over the video if you wish to in order to explain your software & what you got working or not working. Text annotations can also be used if you wish to explain.
  4. Your video **must** be set up without permissions/privacy so that I can open it and view it. To test this, open your video using Chrome's Incognito mode. If you can open it in Chrome's Incognito, I can also open it.
  5. Do not set up any passwords or extra permissions on your YouTube video. It should be publicly viewable without passwords.
  6. If your video is "unpublished", only the people with the link to your video will be able to view it.
  7. Your demo video can and should be short and concise.
  8. Your demo video does not have to show me your actual code.
     1. In the demo video, I want to see your software running and working on your computer.
     2. The purpose of the demo video is for students to demonstrate that their software is working and running on their computer.
  9. Here is an **excellent example of a software demo video from one of my former students, Sneha Revanur (https://www.youtube.com/watch?v=3U37OZe0J1I&t=139s)** .
  10. Copy your YouTube link in to the Google Doc assignment submission file.

Step 2.4 - Add your GitHub History screen capture image to the Google Doc

- In GitHub Desktop, select the **History** tab to show the git commit's you made during development. Copy a screenshot image (jpg or png) of your GitHub history in to the assignment submission file.
- Students should use screen capture software.
- Students should **not** take a screenshot of their computer screen using their smartphones . Using a smartphone produces homework assignments that are not legible. There are free screen capture software on all computers. For example: **Windows (https://www.youtube.com/watch?v=O_55eg00H-w)** and **Mac (https://www.youtube.com/watch?v=uyII9kZ3SZA)**
- There are numerous ways to get a screen capture. On windows, you can use "Snipping Tool" which is already on your computer. On a mac, **these are directions (https://support.apple.com/en-us/HT201361)** .

Step 3 - Save your assignment Google Doc as a pdf

- Save the Google doc as a pdf. This is normally under File->Download->PDF Document. If you can not convert your google doc into a pdf, there are several dozen YouTube videos that explain this.
- Submit the pdf to Canvas. Do not send or message the Professor a link to your Google doc.
- Only pdfs submitted from Canvas are accepted for points.
- If you have questions, please re-review the assignment for further directions.

Step 4 - Upload your pdf to Canvas

- Upload your pdf to Canvas.
- If students wish to verify Canvas received your pdf submission as you desired, students should try to download their submission from Canvas (after uploading it). If students can download their submission, so can the Professor.

Programming Rubric 200 points

Criteria	Ratings			Pts
Specifications	**120 pts** **Exceptional** The program works and meets all of the specifications.	**80 pts** **Good** The program works and produces the correct results and displays them correctly. It also meets most of the other specifications.	**0 pts** **Needs improvement** Program was not submitted or the program is producing incorrect results, does not compile, or does not work. The program produces correct results but does not display them correctly.	120 pts
Readability	**40 pts** **Exceptional** The code is well organized and very easy to follow.	**32 pts** **Good** The code is fairly easy to read.	**0 pts** **Needs improvement** Program was not submitted or or the code is poorly organized and very difficult to read. Code does not adhere to any coding standards. The code is readable only by someone who knows what it is supposed to be doing.	40 pts
Efficiency	**40 pts** **Exceptional** Code runs very efficient without sacrificing readability & understanding.	**32 pts** **Good** The code is fairly efficient without sacrificing readability and understanding.	**0 pts** **Needs improvement** Program was not submitted or the code is unnecessarily long. Code performance is inefficient. Code utilizes brute force techniques when it is unnecessary.	40 pts
				Total Points: 200