MIT App Inventor - Soundboard Project

Quick Links:

Introduction

In this lesson, you will be creating a Soundboard app on MIT App Inventor. With the app, the user will be presented with a list of sound effects to choose from, and if the user clicks on one of the items in the list, the respective sound effect will play.

The sound effects used in this writeup are meme sound effects from mid 2010s. However, the general framework of the application can work with any sound effect files the programmer would like to use.

After completing this lesson, we encourage you to customize the application to your liking, including changing the design, adding more sound effects, etc.

The Lesson Goals for this project are for students to learn how to:

- Use event handlers
- Program with the ListView component
- Incorporate sound files into an application
- Write code with the help of lists
- Practice writing effective and manageable code

Getting Started

We will provide you with the initial designs for this application, which includes the sound files, the main screen and its components.

To start, use the following link to go to the Google Folder where you can find the starter code and other files needed for the project:

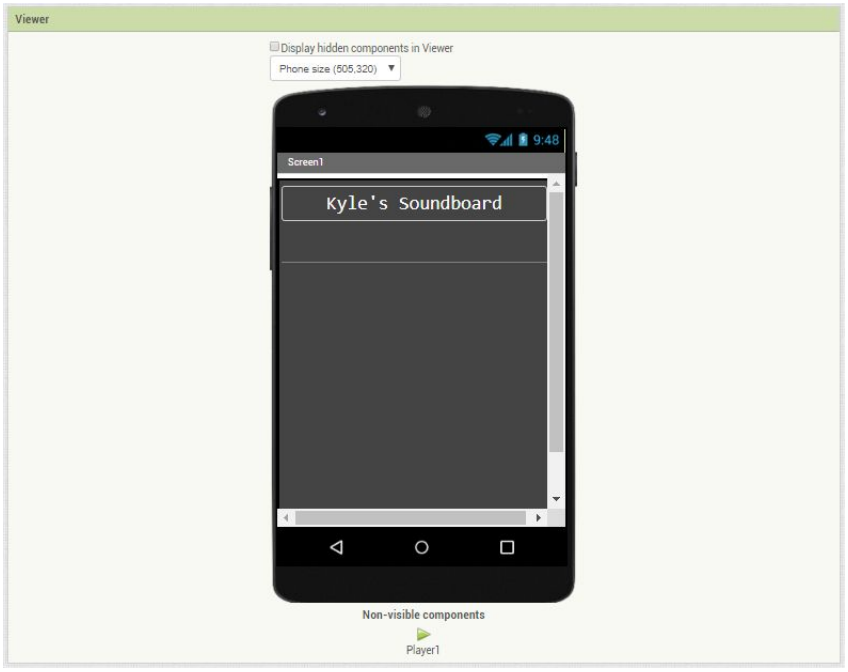
https://drive.google.com/drive/u/1/folders/1p1IoR1XlCbJmZbFsdmu_RyQLkLRS5FlD

The starter code is in the file **Soundboard_Starter.aia**. Download this file on your computer.

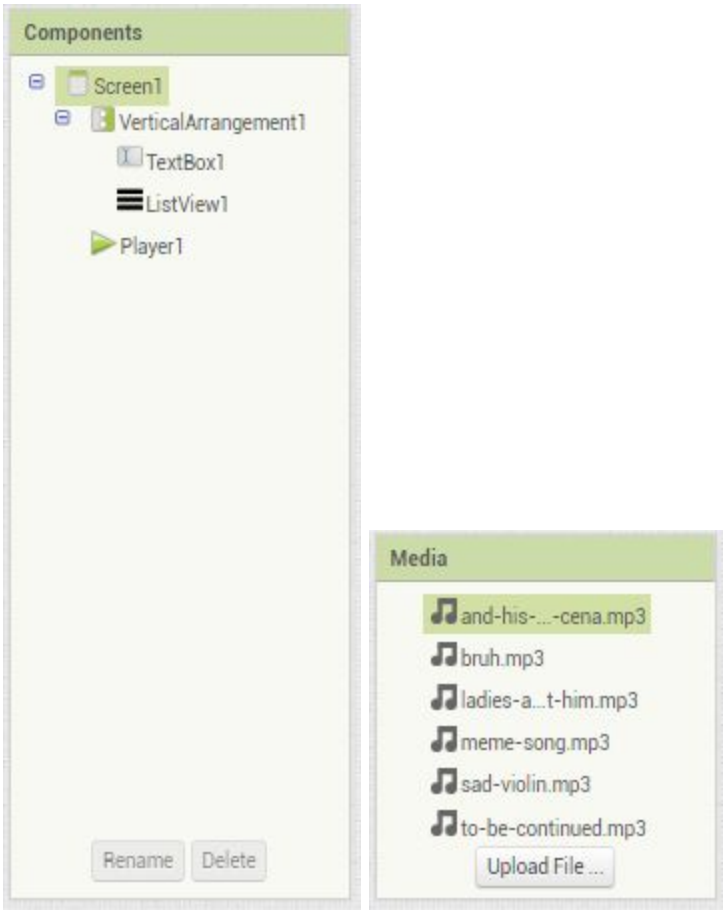
To import the starter code into MIT App Inventor, follow the steps below:

1. Go to the website https://appinventor.mit.edu/. On the top left, click **Create Apps!**
2. Sign in if prompted to.
3. After logging in, click on **My Projects** dropdown menu.
4. From the dropdown menu, select **Import project (.aia) from my computer...**
5. Click on **Choose File** and select **Soundboard_Starter.aia** that you downloaded earlier.

If you do the above steps correctly, you will be shown the main display of the Soundboard Application. Here is what it should look like:



Viewer



Components and Media Section

The design and sound files have been created for you. However, after you finish this lesson, feel free to customize this app to your liking.

The Components

Please read the following carefully as we explain each component at a time. Knowing what each component does will help you immensely in completing this lesson.

Screen1 - This is the main screen of the application. It will contain all the components that the user will see when they first open the app. All components that we add for this application should be put in Screen1.

VerticalArrangement1 - This component allows us to organize our components from top to bottom as opposed to left to right. Here, we can see from the Components menu that VerticalArrangement1 has two components: TextBox1 and ListView1. VerticalArrangement1 is the container that holds these components, and allows us to put TextBox1 at the top of the screen and ListView1 right below it.

TextBox1 - This holds the main title text you see at the top of the screen. In the starter code, it should say “Kyle’s Soundboard.” You can customize this to be your own name later on. But for now, you can leave it be.

ListView1 - This will contain all the sound effect options that the user can choose from to play. You may have noticed something about this component. When you click on ListView1 in the Components section, you’ll see that it’s empty! This is actually done on purpose. What we will do later on is add code so that the sound effect options appear on the app.

Player1 - This is a non-visible component that will play our sound files. Whenever we want to play a certain mp3 file, we will first have to load the source file, then call “Play” in order to play the actual sound. We will create code so that this can work for any sound file the user selects.

Now that you have a basic understanding for how the components work, we can now start coding!

Coding

To start our coding environment, click on **Blocks** on the right-hand corner of the screen.

This will take you to an empty program space. Now, we can start putting code into this space for our application to run successfully.

We will show you how to write the code each step at a time. At each step, make sure you know what each block of code is doing so you have a better understanding of why this makes the application run as it does.

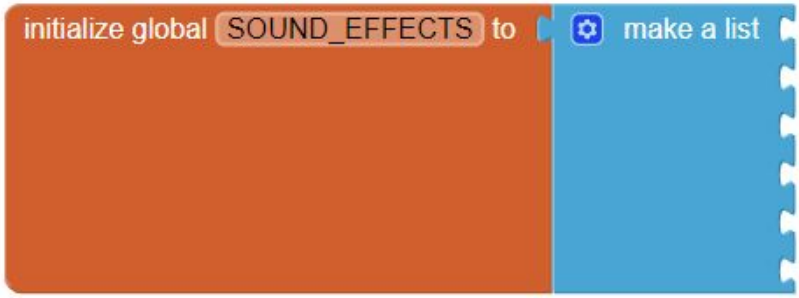
Step 1: Global Variables and Lists

First, in the **Blocks** section on the left side of the screen, select **Variables**, then drag the orange block that says **initialize global name to** to the program space.

Then rename this variable name **SOUND_EFFECTS**.

Next, from the **Blocks** section, select **Lists**. Then drag the blue block that says **make a list** to the tail end of the **SOUND_EFFECTS** block.

In the **make a list** block, select the **gear** icon, then modify the list so that it has **six** items in it.

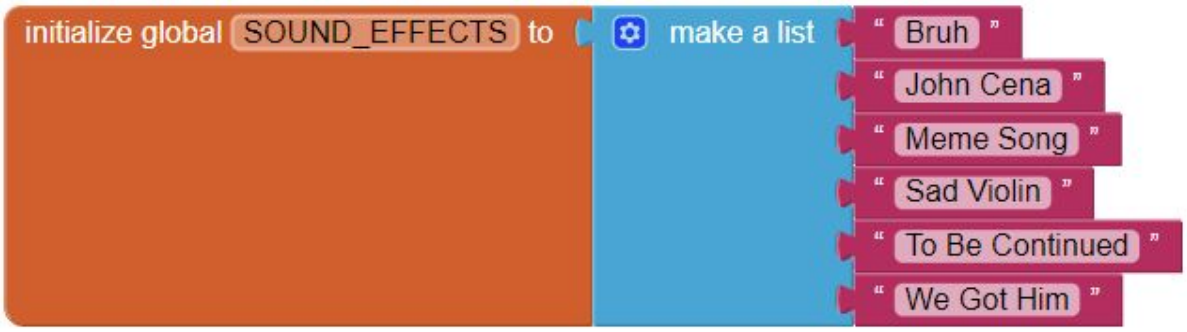


Next, from the **Blocks** section, select **Text**. Then drag the purple block with empty quotes to the end of each item in the blue block list.

Now, we are going to fill the Text blocks with the name of each sound effect. From top to bottom, add each of the following titles to a text block:

- Bruh
- John Cena
- Meme Song
- Sad Violin
- To Be Continued
- We Got Him

If done correctly, your list should now look like this:



Congratulations! We've now finished the first variable. This SOUND_EFFECTS variable is a list of titles that we will show the user as the sound effect options they can choose from.

However, we're not done with making variables yet. We will have to make another variable similar to SOUND_EFFECTS, but this time, it will include the respective media files that will play when a user selects one of the sound effects.

To create this new variable, repeat the steps from creating SOUND_EFFECTS, but this time, name the variable **MEDIA_FILES**. Then, for each item in the list, add the following text (from top to bottom):

- bruh.mp3
- and-his-name-is-john-cena.mp3
- meme-song.mp3
- sad-violin.mp3
- to-be-continued.mp3
- ladies-and-gentlemen-we-got-him.mp3

If done correctly, your MEDIA_FILES variable will look like this:

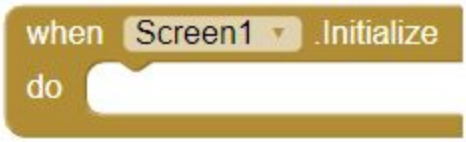
Make sure that the mp3 files in the list are spelled out **EXACTLY** as the files above. If you don't, the application may not work. Double check the spelling before proceeding to the next step.

Step 2: Initializing the List on the Screen

Now that we have a list of sound effect options that the user can choose from in SOUND_EFFECTS, we can now display those options in our application.

If you remember from the description of our components, we said that the ListView1 component is empty when we looked at it. What we'll be doing now is to add code so that when the app runs, ListView1 is now filled with our sound effect options. Follow the instructions below to implement this code.

First, we want to run this block of code when the screen is initializing and loading. From the **Blocks** section, select **Screen1**, then drag **when Screen1.Initialize** to the program space.



Now, when we initialize the screen, we want to modify ListView1.

From the **Blocks** section, select **ListView1** and drag the block the says **set ListView1.Elements to** into the Screen1.Initialize block. (Don't worry about the red X that appears for now).

Next, we specify what elements ListView1 will hold. We will use our SOUND_EFFECTS variable to specify the contents of each element.

From the **Blocks** section, select **Variables**, and drag the **get** block to the tail end of the green block. Then set the value of the **get** block to **global SOUND_EFFECTS**

Now, our app will show the sound effect options to the user. If you want to see the results of the code above, move on to the next step (Step 2.5) to run the emulator. Otherwise, if you want to continue coding, skip to Step 3.

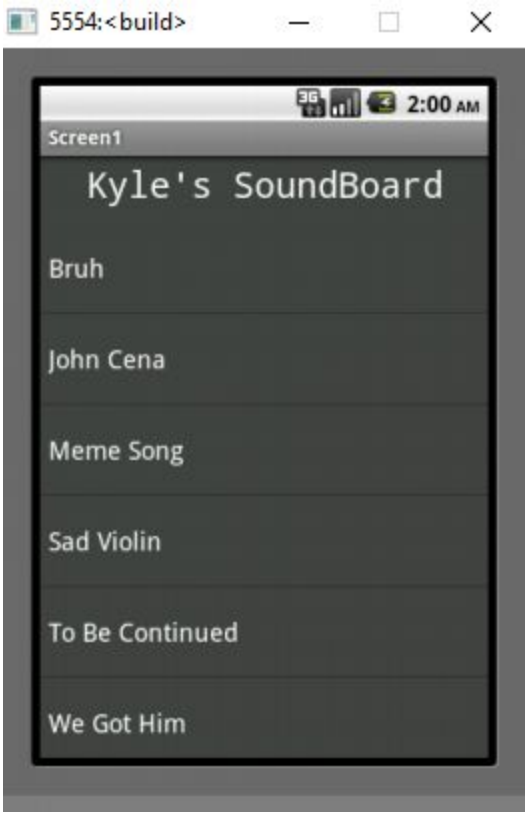
Step 2.5: Running the Emulator (Optional - for now)

Click the following link for instructions on how to set up the emulator:

https://appinventor.mit.edu/explore/ai2/setup-emulator.html

Follow the instructions on the website, and open the application on the emulator.

If done correctly, your emulator should show this screen:



As you can see, the elements from SOUND_EFFECTS now show up on the app. However, when a user tries to click on one of the buttons, nothing happens. We will fix this in the next step.

Step 3: User Selection and Playing Sound Effect

When a user presses one of the sound effects in our app, we want the app to play the appropriate sound file. We will be using ListView1's **AfterPicking** event listener to help us achieve this functionality.

From the **Blocks** section, select **ListView1**, then drag the block that says **when ListView1.AfterPicking** to the program space.

Next, from the **Blocks** section, select **Player1**, then drag the block that says **set Player1.Source** into the AfterPicking block. (Don't worry about the red X that appears for now).

So what do we set the Player's source to? We're going to have to use the ListPicker to tell us what the user chose, then based on what the user chose, we play the correct sound file.

We will explain what the code below is doing. But for now, try to copy it into your coding environment. Now that you have some familiarity with the coding environment, you will be finding the appropriate blocks to fill in yourselves.

Let's take a moment to understand what this block of code is doing. If you are able to understand this code, you are on track to becoming a successful programmer!

So how does this work? Let's say for example, that the user selects "John Cena." The green block that says **ListView1.SelectionIndex** will retrieve the index that "John Cena" is located at. Reminder: the index is its position on the SOUND_EFFECTS list. In this case, "John Cena" is at index 1 (the first position in the list is index 0, the second position is at index 1, where "John Cena" is located).

Next, the blue block that says **select list item** will go into the MEDIA_FILES list and access the item that was at the index of "John Cena". At index 1 of MEDIA_FILES, we see that the item is "and-his-name-is-john-cena.mp3."

So now, we take this mp3 file and load it as Player 1's source. And that's it! Afterwards, once the player has been loaded, we now play the sound file through the app.

Step 4: Stopping the Player

The last step in our program is to stop the player after the sound effect is done playing. Although this step is not necessary for the app to work, it is good practice to ensure any components that are not being run or used are not taking up computation resources in our app.

This problem is mostly for larger apps, but for the sake of practicing good programming habits, we will be doing this in our project.

For this step, you will be the one programming the player to stop when it's done playing. Try to add in code that, when the player is now complete, the player will stop. If done correctly, your block code should look something like this format:

See if you are able to find the correct block of code for this step. Once you have it down, you have now completed the application!

Step 7: Running the Emulator

To run the emulator, please refer to Step 2.5. To quick track, you can click here.

Step 8: Customization (Optional)

Now that you have the basic code down, you are now free to customize this app however you want!

First, if you want to change the design of the app, click on **Designer** in the upper-right hand corner of the coding environment.

Then, to modify a component, click on one of the components in the **Components** section, then modify one of the properties in the **Properties** section. If you have any questions about any of the properties of a component, feel free to ask your instructor or search online.

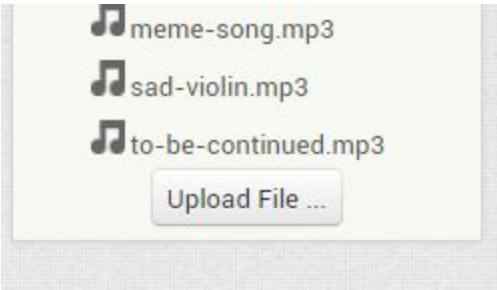
Now, if you want to add more sound effects to the application, what you'll have to do is download an mp3 file then upload it to MIT App Inventor.

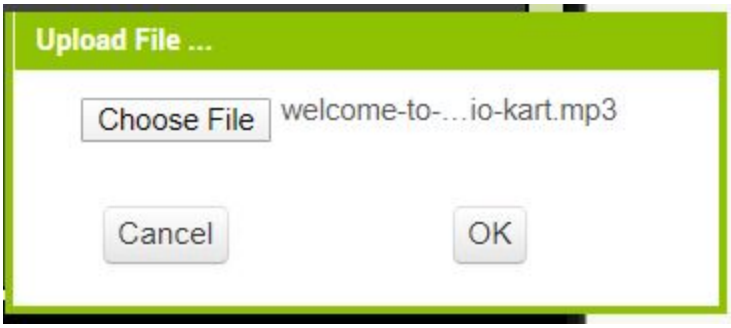
Once you have an mp3 file you would like to add, download it to your computer, then follow these steps to upload it.

1. Click on **Designer** at the right hand side of the interface.
2. Scroll down to the **Media** section and click **Upload File...**
3. Select the mp3 file you would like to use and click **OK**

After uploading it, we then need to update our SOUND_EFFECTS and MEDIA_FILES with the new file so it shows up on our app.

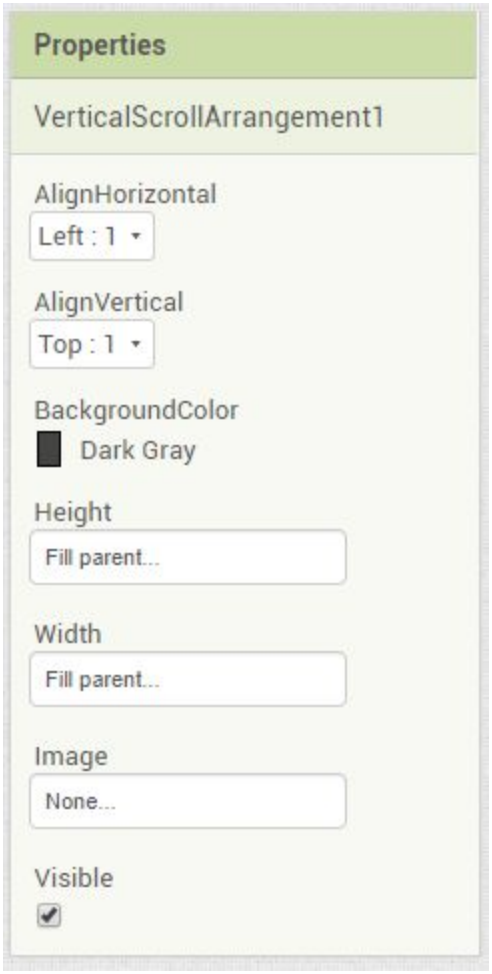
Example: Let's say we have a new sound file called welcome-to-mario-kart.mp3 (The file is in the Google Drive if you would like to follow along). We would upload the file to App Inventor like so...





Then, we add the sound effect to our list (IMPORTANT - Make sure that the indexes between the sound effect name and its mp3 file are the same):

There's one more thing we have to do. Because we ran out of room for a new button, we have to turn our **VerticalArrangement1** into a **VerticalScrollArangement1** component. I will leave this as an exercise to do on your own, but if you do it correctly, the design should look like this:



And voila! Our application is now complete!

Final Remarks

After completing this application, you now have the skills to incorporate sound effects into an App Inventor Application, and have a basic understanding of creating list variables. You have also learned how to program with event handlers, such as when a user selects an item from the list and when a Player stops playing a sound.

Hopefully, this project can give you a basic introduction to App Inventor and its capabilities. I hope you had as much fun creating this app as I did!

Best of luck on your future coding endeavors!